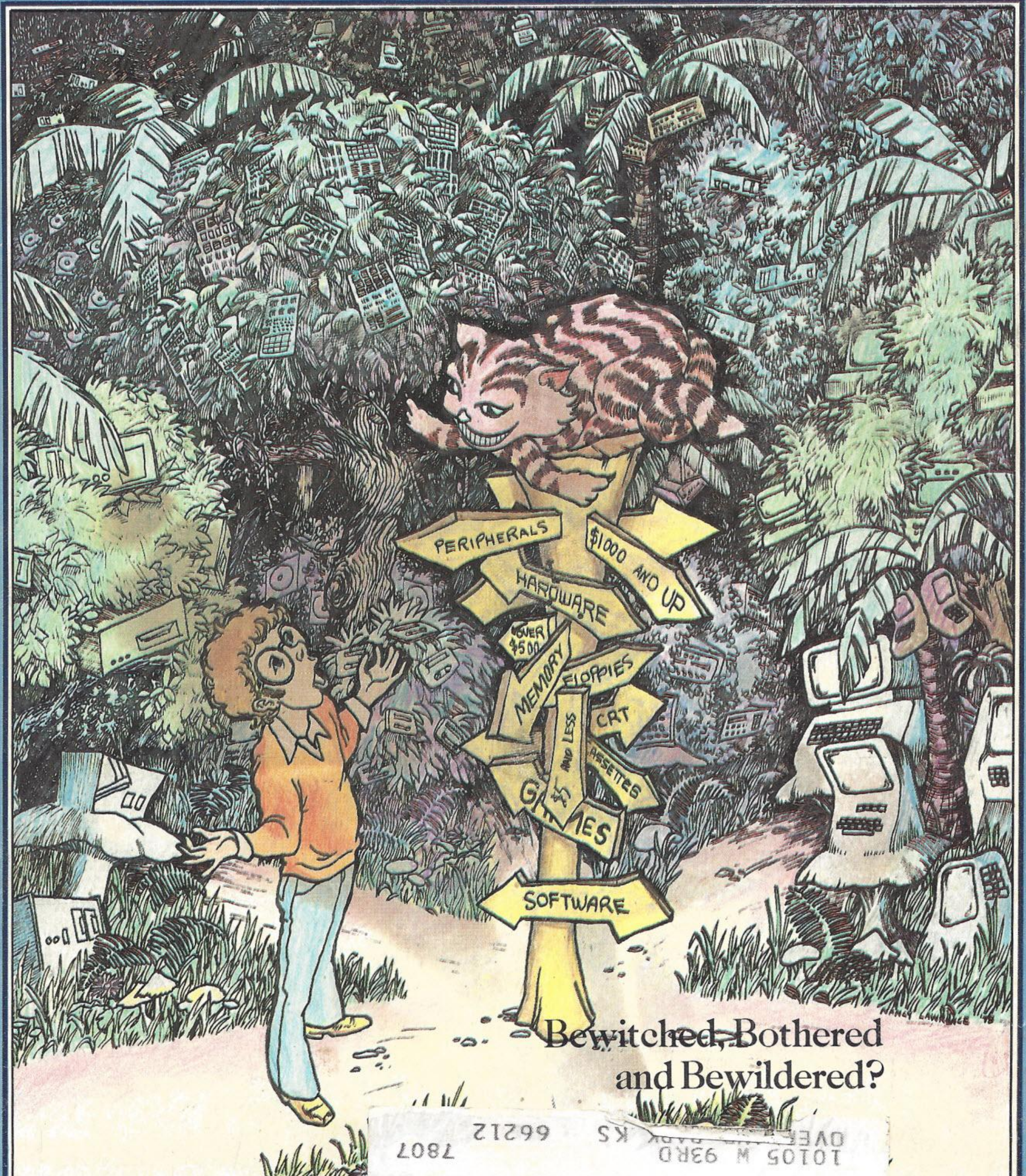
$2.00

Personal Computing

FEBRUARY 1978



Bewitched, Bothered and Bewildered?

Investing in the Stock Market • Computer Graphics

Before you buy a microcomputer, think about service.

At The Computer Mart, we back you with a fully staffed and equipped service department. Trained and experienced in microcomputers. We even have our own "Computer Doctor" ready to diagnose and cure computer problems.

We believe service is an important part of a sale. In fact, our service starts before you buy. We do our own assembly of kits and systems. Then our technicians inspect and test each. Most important, we're there to help you in selecting the right microcomputer for your needs. There to answer questions about compatibility, hardware, peripherals, software and specific applications. Whether your microcomputer is for personal use or for business. Like word processing, control, simulation or data acquisition. When you have a question, we even have a special phone number to call (714) 633-4634. For prompt answers.

At The Computer Mart, our service starts before you buy. So, stop in or call today. Even if it's only to ask a question.

THE COMPUTER MART

633 West Katella Avenue, Orange
California 92667, (714) 633-1222

Hours Monday–Friday 10:00–8:00
Saturday 10:00–5:30

BankAmericard and Mastercharge Accepted, Leasing Available.

At The Computer Mart, we do more than just sell computers.

For professional advice before choosing your system, call The Computer Doctor.

New! **SELECTERM** SYSTEM 9710

The SELECTRIC II* Printer you can TRUST

BECAUSE . . . It's brand new, and fully assembled and tested.

BECAUSE . . . After extensive engineering design and testing by Micro Computer Devices, IBM Corporation has approved the SELECTERM for use with your microcomputer, and provides you with their factory warranty and yearly service agreement for the typewriter. In addition, the electronics conversion portion is fully warranteed by Micro Computer Devices.

BECAUSE . . . You can connect the SELECTERM to your computer within minutes of taking it out of the carton.

IT'S THAT EASY! AND THAT RELIABLE!

FEATURES
- Complete ASCII character set in supplied element.
- Full upper, lower case alphanumeric characters.
- Tab Command, Index (vertical tab), Backspace, Bell—all under computer control.
- Parallel Interface, standard.

ALL ELECTRONICS INCLUDED
- Power supply, electronics and cable sets included to permit immediate connection to the parallel port of any computer, at standard TTL level.

SOFTWARE
- All necessary conversion software in PROM to handle ASCII input, directly.

PRINTER or TYPEWRITER
- May be used as a standard typewriter when not in use with your computer.

OPTIONS
- Dual Pitch, $100
- Correction Feature, $100
- Tractor Feed Platen, $250
- Noise Reduction Feature, $40

AVAILABLE SOON
- RS-232 Interface

PRICE and DELIVERY
- Assembled and tested, $1650
- Available ONLY from authorized dealers.
- Delivery 1 to 2 weeks from receipt of order.

SELECTERM

micro computer devices inc.
960 E. Orangethorpe, Bldg. F
Anaheim, California 92801
Telephone (714) 992-2270

*Registered trademark of IBM Corporation

"Innovators to the Microcomputer Industry"

CIRCLE 4

THE COMPUTER FACTORY
NOW OPEN TO SERVE YOU BETTER

ONLY $595

Yes! It's hard to believe. The PET, a full-fledged personal computer by Commodore.

What do you get? Full ASCII keyboard, 9 inch CRT and tape cassette all in one light-weight unit. Fully-programmable in BASIC (20% faster than most other BASICs).

14K ROM (Read Only Memory). 4K or 8K RAM (Random Access Memory), RAM expandable to 32K. PET's very own graphic instruction set.

For HOME/OFFICE/GAMES
All for just $595 (4K) or $795 (8K).
Portable, Affordable, and Unbelievable.

Apple II is a completely self-contained computer system, with Basic in ROM, full ASCII keyboard in a lightweight molded carrying case. 4K RAM expandable to 48K. But there's more.

Programming in 15 living colors. Apple II's unique color graphic commands allow you to create imaginative color displays. Play pre-recorded paddle games – or invent you own.

Apple II – the personal computer with color.

8K	List price: $1391.75	Sale price: $995.00
16K	List price: $1698.00	Sale price: $1295.00

Must act now! – sale ends *soon*
Save $400

The $5,995 DP Center IMSAI's VDP-80

The Video Data Processor – a complete computer, intellegent terminal and megabyte floppy disk mass storage system. All in one compact cabinet. For just $5995. A complete desk top DP Center. Choose your own printer and get up to 300 lines per minute.

Write or visit us at The Computer Factory and learn how you can utilize the VDP-80 in your own small or large business applications.

NOW AVAILABLE!!! StreetWare™ from National Corporate Sciences, Inc.

The complete cassette library of financial programs for the Commodore PET Computers.

- **Securities Analysis**
- **Options Analysis**
- **Bonds**
- **Mortgage**
- **Annual Report Analyzer**
- **Checkbook Management**

Send for more information.

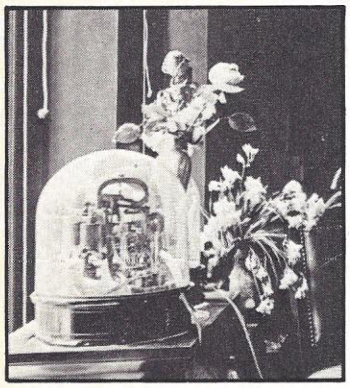
THE COMPUTER FACTORY

790 MADISON AVENUE, NEW YORK, NY 10021
(212) 249-1666 or (212) PET 2001 T-F 10-6 Sat. 10-4

CIRCLE 5

February 1978

VOLUME II, NO. 2



STOCK MARKET PLUNGE 36

A LESS HOME 66

COMPUTER FOOTBALL 80

Cover illustration by Nancy Lawrence

DEPARTMENTS

LAUNCHING PAD

ON THE LIGHTER SIDE

IN THE MONEY

DIGGING IN

Rich Man, Poor Man, Merchant, Physician, Teacher, Lawyer, Student, Musician....

There's an iCOM Floppy Disk System for Everyone!

More Uses
People from every walk of life are adding iCOM® Floppy Disks to their microcomputers for such diverse tasks as payroll, inventory control, mailing lists, game playing, record keeping, parts ordering . . .

We've uncovered some innovative applications, too: The sailboat architect who puts equations and algorithms on an iCOM disk to test his nautical theories; the student who has automated a bowling alley; the iCOM dealer who designed an environmental control system for a university.

More Speed
These users have found iCOM floppies to be much faster and more versatile than cassette or paper tape. With iCOM, programs can be loaded in seconds; files updated in minutes; hundreds of programs can be stored on a single disk.

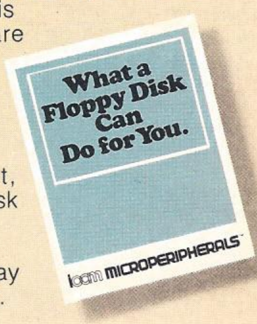
More Models
iCOM has Frugal Floppies™, Dual Floppies, Microfloppies™ (using the new 5¼" diskette), and other new approaches to floppy disk systems. Each is hardware and software compatible with Altair™, IMSAI, Poly 88, Sol-20 and other microcomputers using the Altair S-100 bus format.

More Software
Then there's iCOM's famous software: Powerful field-proven FDOS-II with macro-assembler, string-oriented text editor, and file manager. Plus easy-to-use compatible 8K Disk BASIC. Each with super features such as: named variable length files, auto-file create, open and close, multiple merge and delete . . . and more.

More Backup
We've been building floppies for microcomputers for more than 3 years. Long before the rest. Thousands of systems are operating perfectly in the field. And we're part of Pertec Computer Corporation, one of the largest manufacturers of peripherals, microsystems, data entry products and data processing systems. We'll be around whenever you need us.

More Dealers
Maybe not in quantity, but in quality. We've chosen our dealer network carefully to assure you of assistance every step of the way. Our prices are right. Our delivery is fast. Our dealers are experienced and knowledgeable.

Must Reading
Our free booklet, "What a Floppy Disk Can Do for You" is must reading. Send for yours today or visit your dealer.



icom® MICROPERIPHERALS®

a division of Pertec Computer Corporation

6741 Variel Ave., Canoga Park, CA 91303 U.S.A.
Tel. (213) 348-1391 TWX 910-494-2788

Personal Computing

FEBRUARY 1978 VOL. II, NO. 2

Publisher/Editor
Harold G. Buchbinder

Editorial Staff
Harry Shershow
Jeff Spirer
Ellen Stein
Donald Wood

Art Director
Jane Higgins

Art Staff
Rene Stawicki
Michael Barisano
Richard Sarno

Editorial Production
E. Storm
Sarah Jewler

Advertising Production
Joe Ingram
Merrie Buchbinder

Composition
Nancy Aldrich
Sally Anderson
Paul Cioto

Circulation
Regina Harrington
Joe Gillis

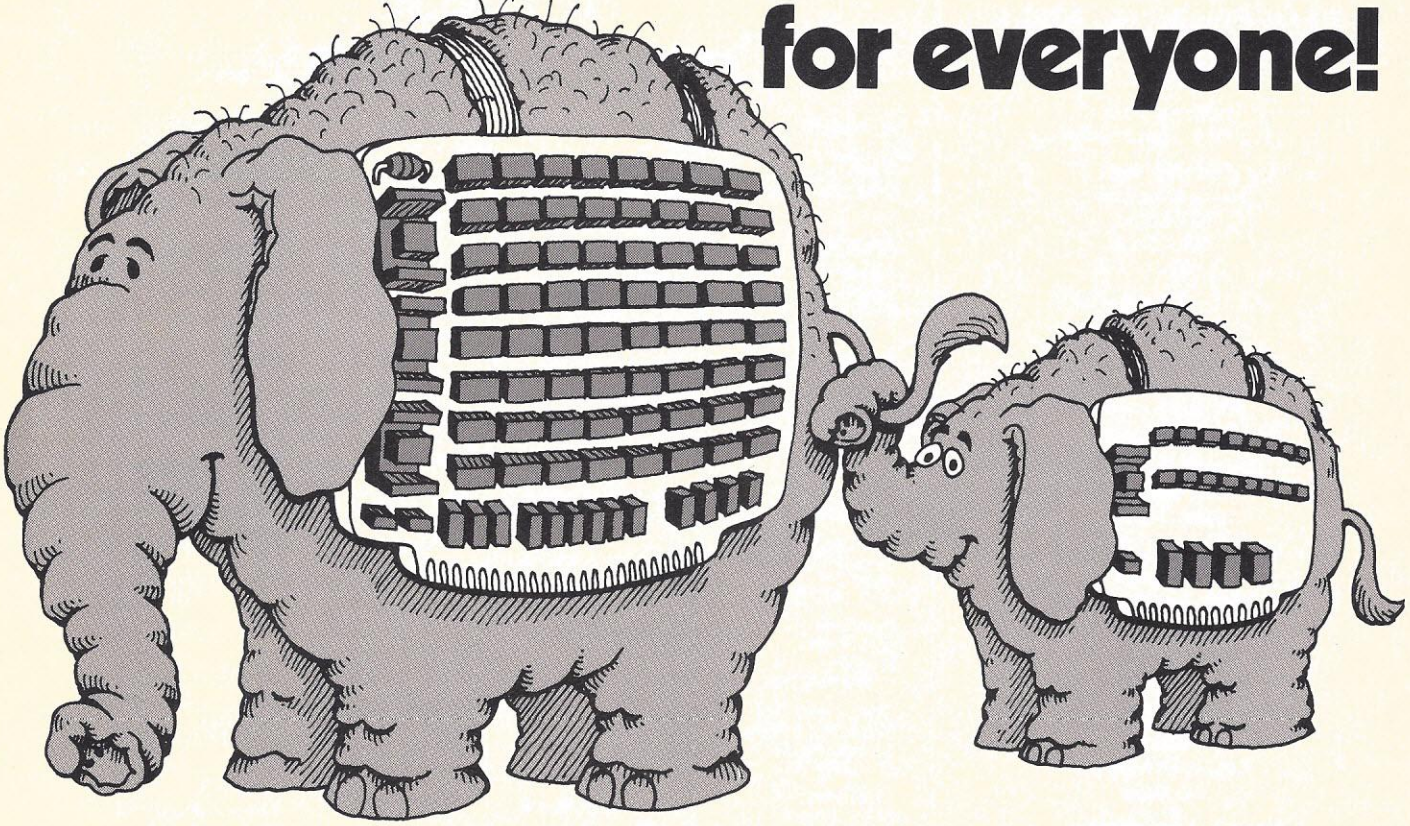
General Administration
Jonna Yager
Esther Shershow
Marion Pearlman
Sarah Binder
Marjorie Morse
Charles Vigilante

Subscription rates. U.S.: 1 year (12 issues) $14; 2 years (24 issues) $26; 3 years (36 issues) $38. Canada & Mexico: add $4/year for surface mail, $8/year for airmail. In Japan/Asia: Personal Computing, CO Magazine, 11-8, Narita-Higashi 1-Chome, Suginami-KU, Tokyo 166, Japan. All other countries: add $8/year for surface mail, $36/year for airmail. Send subscription orders and changes of address to: Circulation, Personal Computing, 1050 Commonwealth Ave., Boston, MA 02215. **Back issues.** U.S.: $2. Canada & Mexico: $3. All other countries: $5.

Publisher. Published monthly by Benwill Publishing Corp.: Harold G. Buchbinder, Chairman of the Board; George Palken, President; Esther Shershow, Treasurer. Executive, Editorial and Subscription Offices: 1050 Commonwealth Ave., Boston, MA 02215. Controlled Circulation postage paid at Long Prairie, MN. Membership in Audit Bureau of Circulation pending.

◄CIRCLE 6

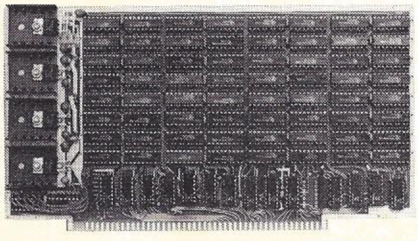
Now, Artec has an expandable elephant for everyone!



8K-32K of static RAM memory. Fully assembled or in kit form.

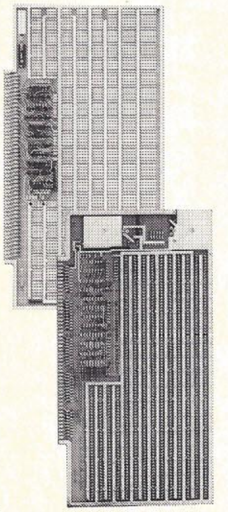
No matter what your needs, Artec has a memory board for you. You can start with 8K of TI 4044 memory on a 5.3″ x 10″ card and work your way up to a full 32K in 8K increments. The access time is only 250ns. The memory is addressable in 4K blocks and is perfect for S100 and battery augmented systems. The Artec 32K Expandable Memory has four regulator positions, bank select and plenty of room for all necessary support hardware. It uses less than 1 amp per 8K of memory (3.9 for 32K), and only +8 volts.

For five years Artec craftsmanship and reliability has been proven in tough industrial use. Now, you too can enjoy breadboards and memories that will work time after time. Send for an Artec board, your order will be sent the same day as received.



Kit: Board &	Fully Assembled Board:
8K of memory—$ 290.00	8K—$ 440.00
8K add on kits—$ 255.00	16K—$ 695.00
Full 32K board—$1,055.00	24K—$1,100.00
	32K—$1,205.00

GP100—$20.00
Maximum design versatility along with standard address decoding and buffering for S100 systems. Room for 32 uncommitted 16 pin IC's, 5 bus buffer & decoding chips, 1 DIP address select switch, a 5 volt regulator and more. High quality FR4 epoxy. All holes plated through. Reflowed solder circuitry.



WW100—$20.00
A wire wrap breadboard, similar to the GP100. Allows wire wrap of all sizes of sockets in any sizes of sockets in any combination. An extra regulator position for multiple voltage applications. Contact finger pads arranged for easy pin insertion.

Buffering Kit—$12.65
All the necessary components to bootstrap any Artec board into your system. Buffering I/O, DIP switch heat sinks and every support chip you need.

TO ORDER: Use your Mastercharge or BankAmericard. Or just send along a money order. We can accept only U.S. currency. Please include $3 handling on all orders. California residents add 6.0% sales tax.

FOR MORE INFORMATION: For more information about these or any of Artec's complete line of circuit boards or for either industrial or personal use, please call or write. A catalog will gladly be sent.

Please send me: (Include Quantity)
___ 32K ___ GP100 ___ WW100
☐ I've enclosed a money order.
☐ Mastercharge No. ______ ______ Exp. Date
☐ BankAmericard No. ______ ______ Exp. Date
Name ______
Address ______
City ______ State ______ Zip ______
Calif. Res. add 6% Sales Tax *$3.00 Handling Encl.*

10% discount for students & computer club members.

ARTEC ELECTRONICS, INC.
605 Old County Rd., San Carlos, CA 94070
(415) 592-2740

CIRCLE 7

FEEDBACK

Loops structure

Dear Editors,

Your readers may be interested to know that the "Towers of Hanoi" game is rather more structured than might appear from the flow-charts in your article on "Loops" in the July/August issue.

The moves are predetermined precisely by the binary counting system, and can perhaps best be illustrated by a simple example based on three disks. The numbers in the column headings, viz 1, 2, 4 and so on, are allocated to the disks, commencing with the smallest; higher numbers being allocated to progressively larger disks. Note too that a leftward arrow signifies a move clockwise, and a rightward arrow a move counter-clockwise.

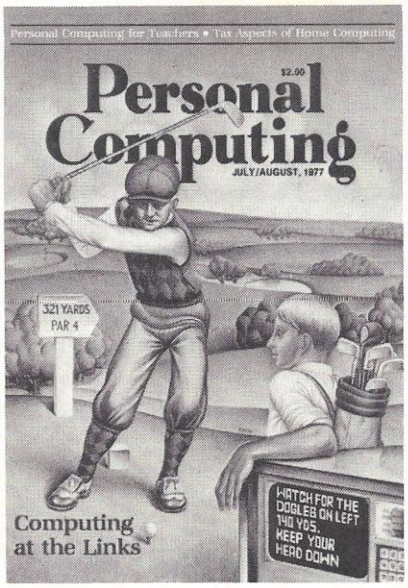
Moves are made on a 0-to-1 transition between rows. The first move,

Move No.	0	1	2	3	4	5	6	7
← 4	0	0	0	0	1	1	1	1
→ 2	0	0	1	1	0	0	1	1
← 1	0	1	0	1	0	1	0	1

therefore, takes disk #1 one peg clockwise. Move #2 takes disk #2 one peg ANTI-clockwise; move 3 takes disk #1 yet another peg clockwise, while move 4 requires that disk #4 be moved, also clockwise. Move 5 is clockwise too (disk #1).

It will be seen that disk #1 always moves clockwise, disk #2 always anticlockwise, and so in alternating fashion for as many disks as required, a move only being made if there is a 0-to-1 transition in that disk's column in going from one move to the next.

However, rather than depending on 0-to-1 transitions, it is easier to determine the disk number by merely shifting-right into the carry (or link) bit, and checking for a '1.' At the fourth move, for instance, a '1' would be detected on the third shift, indicating that the third largest disk (#4) would be the one to move, and as the number of shifts is odd the move would be clockwise. This is very much easier



than comparing with the previous move number and checking for a 0-to-1 transition in a particular bit-position.

It goes without saying, I think, that additional disks would bear the numbers 8, 16, 32 and so on, although this sequence could now be changed to the decimal sequence 1, 2, 3, 4 . . . in order to relate to the number of shifts necessary to set a '1' into the carry-bit.

R. Jones
Abbotsford, B.C., Canada

Let's hear it for tiny APL?

Dear Editors:

I happened to see the "Let's Improve BASIC" article in March/April '77 issue.

My reaction was, let's replace BASIC with APL. APL like BASIC , is easy to learn. It is self teaching. Unlike BASIC, APL remains interesting to the user and the power of APL allows a user to go so much farther and faster than BASIC.

BASIC is a simplification of the IF, THEN, ELSE languages such as ALGOL, PL/I and PASCAL. They are based on a stock machine which does one thing at a time and is therefore limited in power. Recursion is it's only strength.

APL also supports recursion but its greatest power is in its handling of arrays, an array machine can take advantage of parallism. It can do many things at a time. APL is concise. Relatively few keystrokes will solve a problem. APL never leaves one in doubt about operator precedence as almost all other languages do. It follows the same procedure as our positional arabic number system in its handling of operators. That is, it takes operators strictly right to left, just as we take digits when we add or substract numbers.

Both APL and BASIC have been limited in the past in their I/O capabilities but with IBM's shared variables and Burrough's extended formatting APL now matches most other languages.

I suppose APL would take more memory than BASIC, however, I don't see why a "tiny APL" could not be written just as was tiny BASIC. Anyway, I believe memories for micros will grow larger and cheaper in time.

W.H. Daily
Albuquerque, NM

Oops!

Editors:

I noted a couple of errors in your Zilog instruction set table (Sept/Oct 1977). The opcode for CP L should be BD instead of BC. Also, the opcodes (77 and 7E) for indirect load and store of A (LD (HL), A and LD A, (HL)) are duplicated in the top table and the one below it. Not wrong, but confusing to those not familiar with the 8080 set.

Another point that might puzzle the innocent is 7 of the opcodes (e.g., LD A, A coded by 7F) that do nothing at all, are really NOPs (that in the 8080A takes 5 clock periods instead of the usual NOP 4, while in the Z80 all take 4.)

Nevertheless, the table is very worthwhile, making it very clear that the majority of Z80 opcodes are 2-byte and quite a few are 3-byte.

H.T. Gordon
Oakland, CA

Comments, questions, and criticism are welcomed. All letters should be addressed to 1050 Commonwealth Ave., Boston, MA 02215. All material is subject to editing for length, libel and standards of good taste. Signatures may be withheld upon request.

Radio ham

As a radio ham I am interested in building either a small computer for my personal use at home, or purchasing a ready made computer from a reliable manufacturer in the US.

I'm interested in names and address of firms that market such equipment.

Hans Wenk
4153 Reinach
OB. Rebbergweg 1
Switzerland

Classroom correspondence

For several years we have been running a system of programs to keep track of grades on a teletype and timeshare link. Recently we have had to switch to three Sol-20s. We are now keeping track of 800 students on one Sol-20 equipped with a North Star Disk and 48K of memory. The GRADER package keeps track of advisory attendence, student grades, top students, progress reports to tell parents of a child's poor performance and scheduling of students on the terminal. In addition, 60 students learn elementary BASIC programming. We work on everything from from baseball programs to pong games and two-person Star Trek. We're planning to also run a debate tournament by computer. We would appreciate correspondence from other schools.

Ken Meltsner
c/o John F. Kennedy High School
4300 Cutting Blvd.
Richmond, LA 94804

A second saved is . . .

I have used the Facit Electronic Time Recorder system about thirty days on a trial basis and find it most useful in my affairs. Are there any comparable products on the market?

This particular product was priced at $1,595 this past summer.

Albert W. Hall
No. 11 Dunlap Ct.
P.O. Box 446
Jacksonville, IL 62651

In search of

I'm looking for a small company willing to build a device I've patented — I've already found a company interested in selling it. Initially a few would be ordered but later I believe we could order 100 or more per year.

The device consists of a case, 5VDC power supply, 16-18 DPDT switches, 18-20 seven segment displays, 6-7 triple AND gates, sockets, 144-160 resistors, PC socket or sockets and 1 or 2 large PC boards. I need to know how much the device will cost and will supply additional details.

Arleigh Bruce Hughes
158 Havlani St.
Pukalani, HI 96788

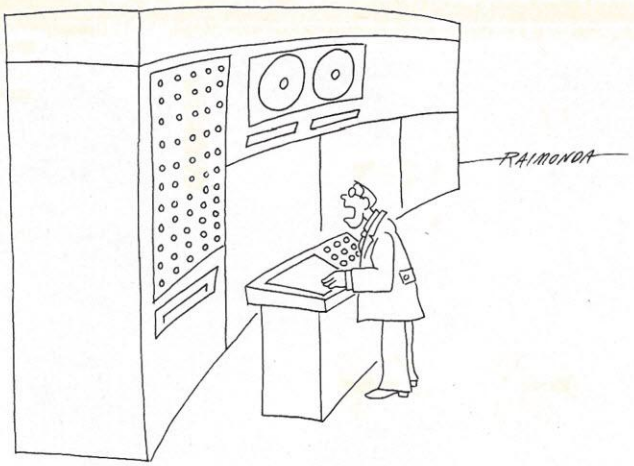
Who's at the top?

I am interested in getting involved in the information storage and retrieval business. What I would like to know is who you would consider to be the most noted designers and firms in the US and what they are most noted for by way of research development. In addition, it would certainly be inspiring and exciting to see articles by such men and/or firms on their present and future projects.

Finally, I would like to know how to begin to enter this field, to know what to learn, and where to learn it.

Bryan C. Greifinger
271-3677th Rd.
New Hyde Park, NY 11040

Gang? We'd also like to hear your suggestions. No fair nominating yourself. . .



"What do you mean you'll answer that question when I'm a little older?

Printer plea

I am attempting to help a friend establish a small data processing system that will be capable of the usual business functions and also do justification for offset printing.

The problem is that the proliferation of small systems and software for them is overwhelming – at least to me as I have been 15 years in large commercial systems and can see little organization in the micro market.

The information I am seeking is a recommendation as to a microsystem to which I might interface a "daisy-wheel" printer that has justification software available for line copy with multiple CRT input. I know the lot of them will do the accounting requirements.

A.K. Smith
FCP, Box 1000
Steilacoom, WA 98388

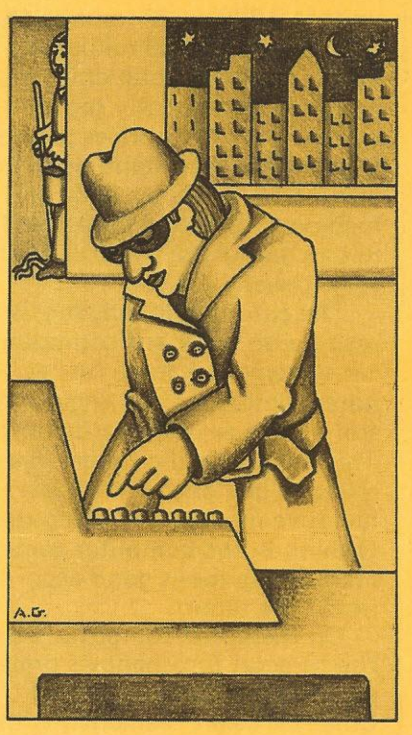
Input/Output material may be sent to Personal Computing, *1050 Commonwealth Ave., Boston, MA 02215. Please include your name and address.*

RANDOM ACCESS

Computer crime

Today's villians don't sport handlebar mustaches and silver Colts in their holsters. Nor do they wear masks and sneak around in the dark. In fact, that pleasant-looking guy sitting across from you at lunch sipping nervously on on his third martini may just have committed a felony — computer fraud.

As computers rise in popularity and use so do computer-related



crimes. A recent *New Yorker* article told the story of one disgruntled computer programmer who programmed his company's computer to add the raise he felt was due him onto his weekly salary checks. His actions went undetected right up until the time he retired. But, as is the case with most novice criminals, he slipped; he forgot to reprogram the computer . . . and the computer continued to process the padded checks after he had left the company. His error was noticed and he was apprehended and convicted.

To keep up with the growth of white collar computer-related crime, the FBI now instructs its agents in the art of detecting computer fraud. Jim Barko, unit chief of the FBI's computer-fraud squad, conducts a 4-week, thrice yearly "Economic and Financial Crimes Training Course" in the FBI's special school at Quantico, Virginia. So far, the agents have been successful in their work, and their batting averages keep rising.

One of the cases the FBI team cracked last year involved the theft of classified files from a Federal Energy Administration (FEA) computer. An FEA employee had discovered a way to use his home computer terminal to remove the classified information from the office's computer. A night janitor in the office caught this computer crook when he noticed something was fishy. Barko's force was informed and they caught and convicted the culprit.

As their manpower increases in the future, Barko hopes to further counter the rising trend of computer crime by expanding the training program.

What's up doc?

The pharmacist at your local corner drugstore may process your next prescription by computer. Now even the smaller pharmacies that handle only 75 prescriptions a day can justify the cost of working with a small minicomputer.

One system, the A-5 small-business minicomputer offered by the Olivetti Corporation, has been specially tailored for pharmaceutical applications. The computer is designed to improve overall efficiency by eliminating many of the repetitive and often menial daily tasks and paperwork.

The A-5 prepares prescription logs and automatically bangs out labels and patient receipts. The computer will also check refill limits of prescriptions, maintain patient profiles, and issue daily recaps of medications dispensed. In addition, the machine performs business transactions and

automatically processes third party billing forms.

This pharmaceutical minicomputer consists of the A-5 model 20 with sprocket and manual front feeds. The unit, which fits on a countertop, has a 4K byte central memory, buffered electronic keyboard, operator guidance display panel, print unit, and magnetic card unit. Modularly designed, the basic configuration can be extended after installation to accomodate on-line communications capabilities.

On a five year lease with option to buy, the system costs approximately $325. The system (hardware and software) fully installed will run about $11,400.

Additional information on training programs, analysis, service, supplies and maintenance agreements for the A-5 are available from: The Olivetti Corporation of America, Pharmaceutical Systems, 500 Park Ave., New York, N.Y. 10022.

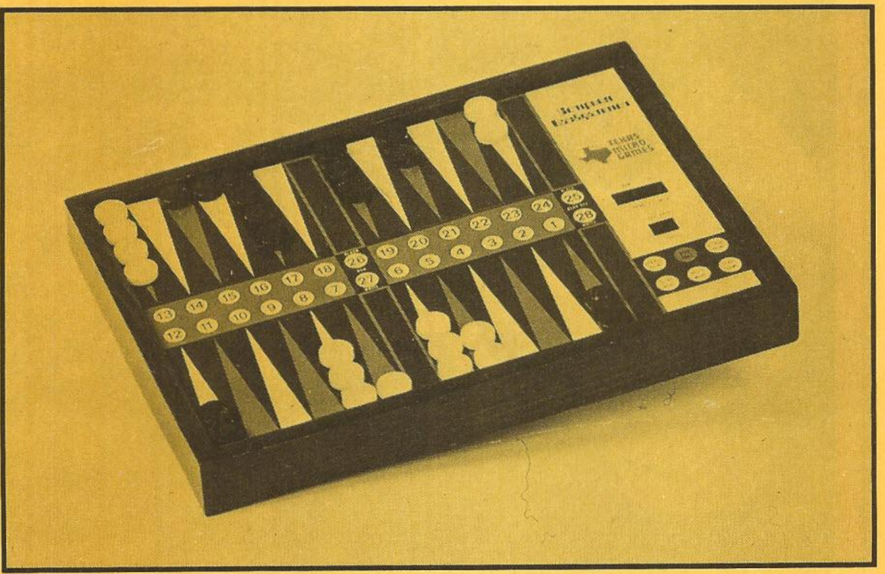
Games People Play

First it was chess, now backgammon. Who knows? Maybe checkers are next. What? Playing it "alone". Texas Micro Games has just come out with a computerized backgammon set. The set fits the needs of the addict in search of a partner at 2 a.m., the gambler who wants to practice without the risk of losing cash, the novice who wants to learn without criticism or the backgammon enthusiast who enjoys a good challenging game any time of day.

The self-contained set operates on standard electric current. All you have to do is plug it into the outlet. Six command keys control the plays. All moves, yours and the computer's, are initiated by depressing the appropriate key. The computer is touch-sensitive, reading each move as it's entered and computing the next.

To begin the game and intialize the board setup simply touch the "Start Game" button. The display will then light up with the word "roll" — you press this to see who goes first. All the moves are determined by the display of a simulated "dice" reading on a digital display; there are no dice in the game. Since the computer computes each move before it makes it, the game is both a teacher and a competitor.

According to the manufacturer, the game will retail for "about $200" and will be available through retail outlets throughout the country. For those of you living miles from the nearest sign of civilization, the game is also available through the American Express or Horschow's Collection mail order catalogs.



In the crystal ball

A number of companies engage in the business of analyzing the trends and predicting the future of the computing industry. In a recent study of the field of microcomputer use in games, appliances, automobiles, home computers and other consumer products, a New York City technological research firm, Frost and Sullivan, warned that there are "as many uncertainties and risks present as there are opportunities." So before you decide to plunge into the field, read on.

The study projects that the worldwide microprocessor and microcomputer market will expand from its stand at 12 million units in 1976 to 22 million this year and on to 130 million by 1984. However, total dollar revenues will significantly lag unit volume growth as prices plumet from where they stand now, at about $10, to $5 by 1984 for high-performance and low end microcomputers; and to $3 for very low-end micros.

The automobile industry is ready with on-board computer car-systems to handle fuel economy, emissions and engine control as well as dashboard controls. This field, says the report, promises to be "one of the most lucrative markets for computer." Growth in the computer game market also looks good according to the report.

On the home computer front, the forecast foreshadows price erosions on kits, boards and other hardware elements, with "the 8-bit microsystems continuing to dominate."

But, the study warns, uncertainties cannot be forgotten. The competition between 4 bit vs. 8 bit vs. 12 bit vs. 16 bit vs. bipolar bit slices vs. special technologies, will wage on. The custom designed computer will confront the general purpose system, the programmable micro will stand up against hybrid compromise designs.

The report also predicts that the rise of the microprocessor

and microcomputer will cause much of the business for "earlier generation integrated circuits and discrete components to completely disappear."

The 174 page study (reference #492, $675) can be obtained by writing to Customer Service, Frost and Sullivan, Inc., 106 Fulton St., New York, NY 10038.

Reporting on the future of the home videoplayer and recorder, Creative Strategies Incorporated, a market and research consulting firm, says that this industry will "fail to make a sizeable penetration of the consumer electronics market over the next few years." The group attributes this to high risks and elusive profits in the industry. But they also say that U.S. retail sales will grow from $90 million to $318 million by 1981. They don't deny this 28% annual growth rate, but they say their "research indicates that substantial penetration of the home market is still years away."

The overall prediction includes a sustained and substantial growth rate over the next few years, with price as the most prohibitive factor preventing penetration of the market. In addition, they see videodisc players facing competition from all forms of home viewing entertainment, especially Pay-TV.

The full report ($595) is available from Creative Strategies, 4340 Stevens Creek Blvd., San Jose, CA 95129.

Flick of a Switch

Imagine this: your alarm goes off at precisely 7 a.m. on a cold winter morning. You rise to find your your bathroom already heated by the electric baseboard heater, a freshly brewed pot of coffee downstairs, a warmed sweet roll in the oven, and you car preheated by a block heater. Did Tinkerbell pay visit during the night? Was a family member suddenly overcome with generosity and good feeling? Or is your Coby I System operating as programmed?

This new microprocessor-controlled remote switching system is programmable for convenience timing and control, burglar prevention, and home energy man-

Keep the presses rolling

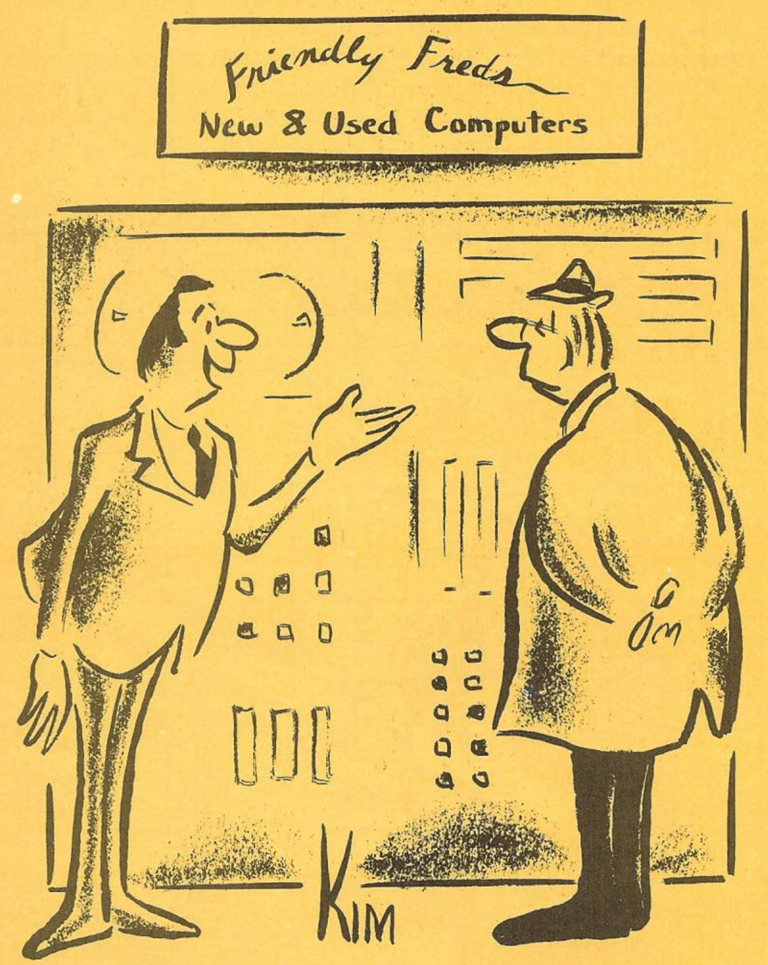
Hobbyists, small business operators and personal computer enthusiasts all have magazines written especially for them. But the computer salesperson, the person who must know the field inside and out, has been neglected.

But starting this February, stories on how to sell, explain and understand computers; how to efficiently manage a computer outlet; and the latest in trends, technology and products will all appear in monthly issues of a new magazine, *Computer Dealer*.

The magazine will feature articles of interest to computer store owners, managers and sales people; consumer clubs and their members; manufacturers and distributors.

Another magazine already on the market, *Computer Retailer*, also deals with this same market.

But while both these publications tailor their material to the needs of the sales-end as opposed to the customer, many of their articles contain information valuable to anyone interested in computers for personal use. And even if an article doesn't help increase your understanding of the field, it might just help you judge how much your salesperson knows.



"This beauty was owned by a little old lady who only used it to figure out her grocery bill."

agement. This means the system can monitor everything from your stereo to your burglar alarm.

The Intel-designed 8085 processor transmits coded digital instructions through standard house wiring to the remote control units.

These terminal units decode the signals and execute instructions. No wiring is necessary; you just plug the system into any standard wall socket.

The system can program devices on an on-off basis, or on short or long cycles ranging from a second to 100 hours. All programs are entered, stored and modified through 12 function and control keys and a 10-key numeric pad with two extra keys for a.m. and p.m. designations.

The Coby I Control unit features solid-state displays that indicate time, date and number of remote units being programmed. Each control unit can direct up to 100 remotes.

The control unit includes 2048 words of low power programmable RAM, 2048 bytes of ROM, the 8085 processor plus associated chips, a power supply cell that doesn't stop running when transfered from room to room and signal-generating hardware.

Presently the remotes are available as switches between plug-in appliances and wall socket for 115 volt circuits, 10-40 amps. They will also be available (in March) as replacements for standard wall switches and as in-line 220V remote controls of built-in appliances.

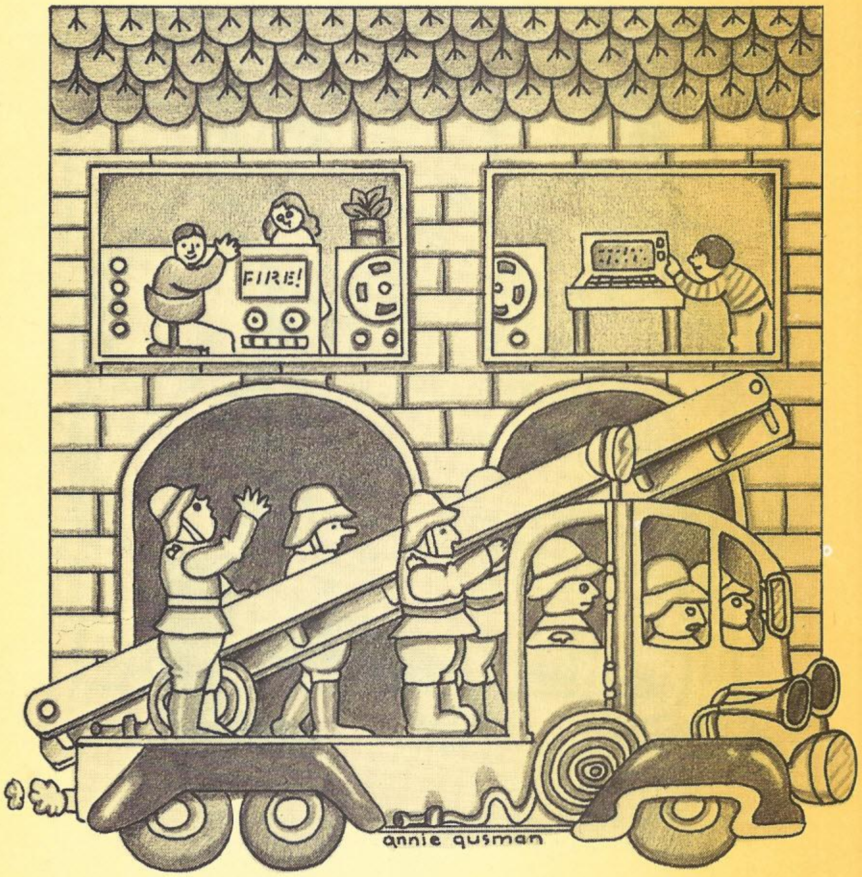
The price on these remotes, for an unspecified introductory period, is $40 for each 10 amp model. For more information you can write to: Energy Technology Inc., 1601 South Main St., P.O. Box Q, Las Cruces, NM.

With a complete system in your home you could turn on the lights, turn off the lights, turn on the television, turn off the television, turn on the heat, turn off the heat . . .

A fire fighter's best friend

A Dalmatian (usually named Spot) used to help firemen rescue people in distress. A keen eye, a set of binoculars, and a 100 ft. tower also used to help fire fighters control forest blazes. Now we have computers on the job, and they don't bark at crowds (or stop at a convenient tree) or suffer from eye strain.

Scientists at the Riverside, California Forest Service Forest Fire Laboratory are using a computer at the local state university to help predict behavior patterns of forest fire blazes.



The problem of predicting behavior patterns of a blaze involve accurate measurement of wind and temperature conditions in the lower atmosphere. When a crew knows humidity, wind and temperature factors for the next six or 12 hours they can then better prepare their defenses.

The scientists are using the computer to store topographic information for two test areas. This data, combined with mathematical models of the lower atmosphere, allows them to convert the upper atmospheric weather predictions (which can be accurately forecast) to applicable ground level forecasts.

When applied to other parts of the country, a series of telemetering stations in test areas will record wind, temperature, and humidity conditions approximately every half hour. These measurements will be transmitted to the main computer nightly to be compared to the atmospheric predictions used to help create the mathematical model.

In a related project called Focus, Forest Service scientists have experimented with various options in allocating fire protection resources. Through a simulation of a fire protection organization set up by a UCLA computer, they practice dispatching forces to a blaze, applying fire fighting

techniques and predicting the probable outcome of the fire.

The project has been successful in allowing fire officials to better dispatch ground and air equipment. Plans are now set to make the project operational throughout the Forest Service area this year.

On a smaller scale, numerous fire departments across the country have replaced their helpful hounds with computer information terminals located in the engine house. One large city utilizing such a system is Kansas City, Missouri.

Now, when Kansas City fire fighters race to the scene of a fire, they are radioed information from the Fire Alarm Exchange's computer concerning residents of the burning building, the owner's name, and whether or not there are guard dogs or other dangerous animals present on the property.

The computer also stores information about firemen on the force. If a fire fighter should be injured on the job, the computer will output his blood type and allergies, whom to notify, what hospital he prefers and the name of his personal physician. The machine also compiles fire reports, maintains information on equipment status and schedules, and pinpoints the locations of equipment.

One problem inherent in the system is its dependence on the public to supply pertinent data. The department would like to obtain information on each piece of property — detailing the location, number of stories, type of structure, and number of people in each building at various times of the day — and then program this material to a manageable size.

But the system has had difficulty keeping its listings updated. For example, one day a computer operator called up a "guard dog" listing for a specific address. The computer informed him that there were "no guard dogs here, but there is a fully grown lion. . ." But when the fire department checked up on this they found the lion had moved. . .

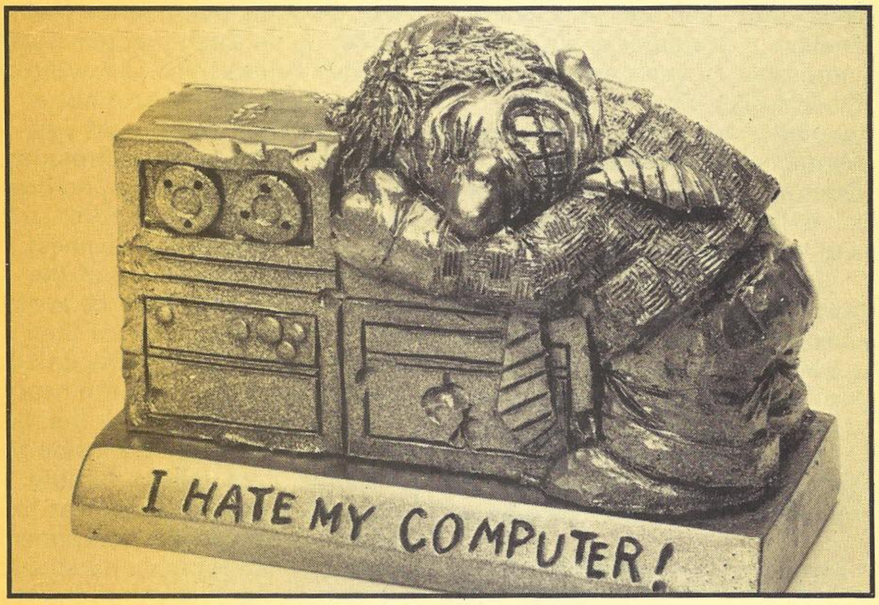
You can work it out

Let's face it. All relationships have their ups and downs. And we know these feelings have to be recognized and dealt with. So if your computer is "down" more than it's "up", perhaps it's time you acknowledged the fact.

You don't have to hide your relationships from friends. Let them know how you feel. Who knows? They may be in the same situation you are. So as a gift, or for yourself, consider this idea for a new computer room decoration/paperweight: a "I hate my computer" sculpture — it might even help your emotional release. Instead of hitting your machine, you can always throw the piece of sculpture across the room.

Or, if your relationship is going well, you might consider a "I love my computer" sculpture.

These sentimental plaster pieces sit six inches long and sport an antique gold finish. The price is $6.95 postpaid, satisfaction guaranteed. Write to Brian Productions, 2949 Southfield Rd., Xenia, OH 45385.



Don't miss 'em

As the interest in computers for personal and business use increases, so does the number of computer shows and expositions. The following sample will give you an idea of what's coming up, but be sure to double check on each show before you travel; some may be canceled at the last minute.

One of the first expositions of the year will be the 1978 Business and Personal Computer Sales and Exposition at the City Line Holiday Inn in Philadelphia, PA, February 27 to March 1. The show will include new equipment and product exhibits and a number of special seminars and workshops. Tickets are $3 if you preregister, $6 at the door. Information on the show can be obtained by writing to Produx 2000 Inc., Roosevelt Blvd. and Mascher St., Philadelphia, PA 19120.

To follow-up on last April's successful San Francisco Computer Faire, a Second West Coast Computer Faire will be held in the new San Jose convention center on March 3, 4 and 5. Some 150-190 exhibitors and an anticipated (conservative) estimate of 10,000 - 15,000 attendees are expected.

The Faire will direct itself toward the computer and elec-

tronic hobbyist, the educator, small business operators and amateur radio enthusiasts. There will be seminars, speakers, exhibits and a presentation of papers. For more on this fair write to: Computer Faire, Box 1579, Palo Alto, CA 94302.

On April 28-30 Percomp '78 will be held at the Long Beach Convention Center. A projected estimate of 20,000 attendees will take advantage of the some 80 free seminars, tutorials, and demonstrations. The three days of conferences will include sessions on ham radio communications, graphic systems, home applications, pattern recognition, programmed learning and word processing. There will also be a number of special business use demonstrations. For information on Percomp '78, write to: 1933 Seventeenth St., Suite 108, Santa Ana, CA 92701.

A special personal computing festival will highlight the 1978 National Computer Conference, June 6-8. The Disneyland Hotel, Complex in Anaheim. CA is the site set for the festival. Hobbyists, consumers, students and computer professionals will attend the product and service exhibits, presentations of papers, and individually-designed displays of personal computing systems and applications. A variety of microcomputer systems, kits, components, hobby equipment, terminals, disk and cassette tape systems, hardware and software will be on exhibit.

Approximately 30 sessions are planned to cover topics including graphics, music systems, speech synthesis, computer games, systems for small business, software design, optical scanning, microprogramming, and more.

One day and three day registrations are available for the festival. For details write to AFIPS, 210 Summit Ave., Montvale, NJ 07645.

The Third West Coast Computer Faire has been set for October 27-29 at the Los Angeles Convention Center. Around 200 exhibitors, and an anticipated 12-16,000 people will attend the Faire. Paper proposals will be accepted until August 28th.

For more information on either the Faire or on submitting your work write to: Computer Faire, P.O. Box 2579, Palo Alto, CA 94302.

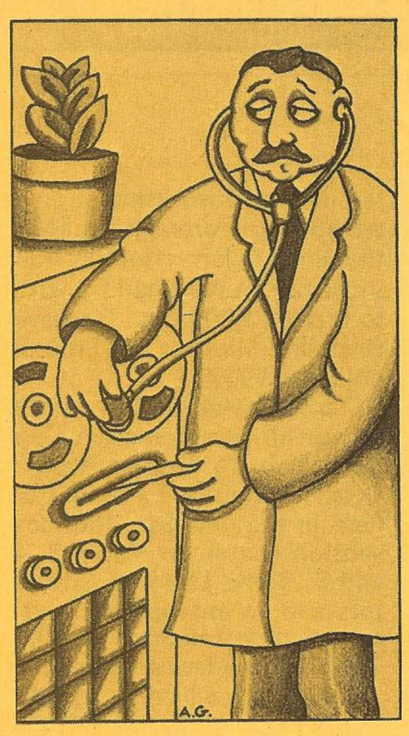
And across the waters, Mini/Micro 78 will feature a special exhibition of mini/micro equipment and systems, in Frankfurt/M, West Germany. Included in the 3-day March 14-17 exhibition will be a symposium on technology, legislation, and commerce. For more information write to: EXHIBIT Ausstellungs GmbH, P.O. Box 174201, D 6000 Frankfurt, Germany.

Is there a doctor in the house?

This doctor doesn't make house calls, but he has been known to give some free advice — to ailing computers. The doctor is Grant Snellen, a computer engineer with more than 15 years experience in the industry. Snellen has just opened a new shop in Cambridge, Massachusetts, The Computer Doctor, Inc. This "computer clinic" offers a full range of repair and counseling services.

Snellen works on a service contract basis where he makes regular checkups on the computers, or on a walk-in basis. Depending on the availability of parts, local service is usually rendered in two days.

So far, the majority of the doctor's time has been spent teaching potential kit builders "preventive medicine". For this Snellen offers a "rebuild-it-yourself kit" for sloppy solderers. The novices visit his shop and practice their soldering on small practice boards. Snellen believes you can avoid a lot of ills as well as turn out a "trouble free" machine if you can master soldering techniques. Luckily for him though, no one has discovered a "computer cure-all" yet — he's still in business.



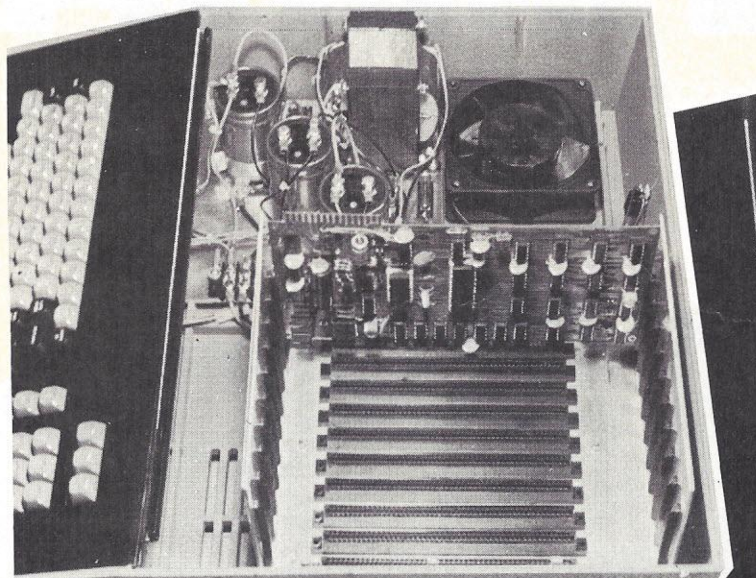
Snellen believes a lot of the problems he sees could be avoided if owners took care of their machines. To do this, the doctor recommends placing a fan near the computer to draw in the dirt that inevitably collects inside the machine. Left unkept it could clog the vents and keep heat from escaping. The plastic components of micros are more vulnerable to thermal breakdown than the sturdier ceramic elements found in more expensive machines.

The computer doctor also recommends plenty of exercise. "Leave your machine on, it's not a light bulb that should be turned off every time you leave the room. Unless you're going away for a week or more, just let it run."

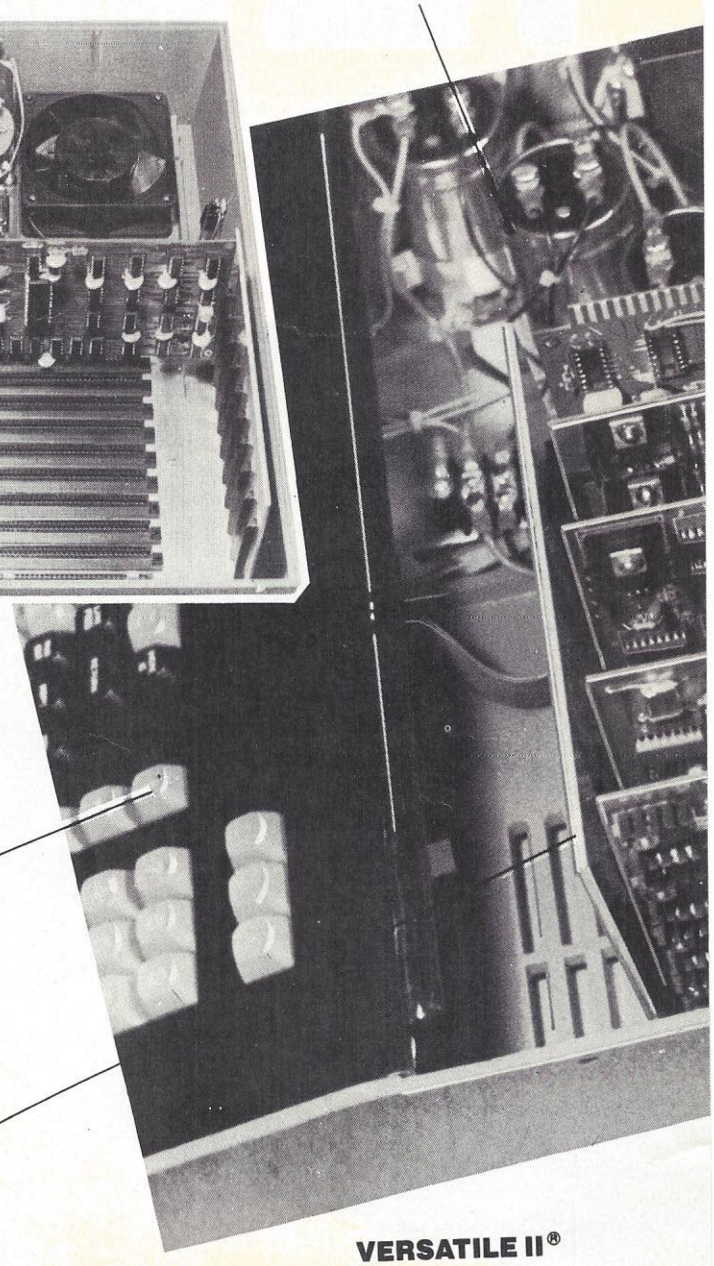
There are certainly more "computer doctors" around waiting to give first aid. If you know of any or if you have advice of your own to give, why not write and let us know. We'll compile a list of emergency numbers that you can paste up near your phone next to "emergency police", "emergency fire" and "emergency doctor". And the next time your computer comes down "with the bug" don't panic — help may be just around the corner.

OR IS

RUGGED AND EXPANDABLE, THE VERSATILE CRT® CAN BE ANYTHING YOU WANT IT TO BE, FROM THE SIMPLEST TERMINAL TO A COMPLETE COMPACT COMPUTER—**THE VERSATILE II®**

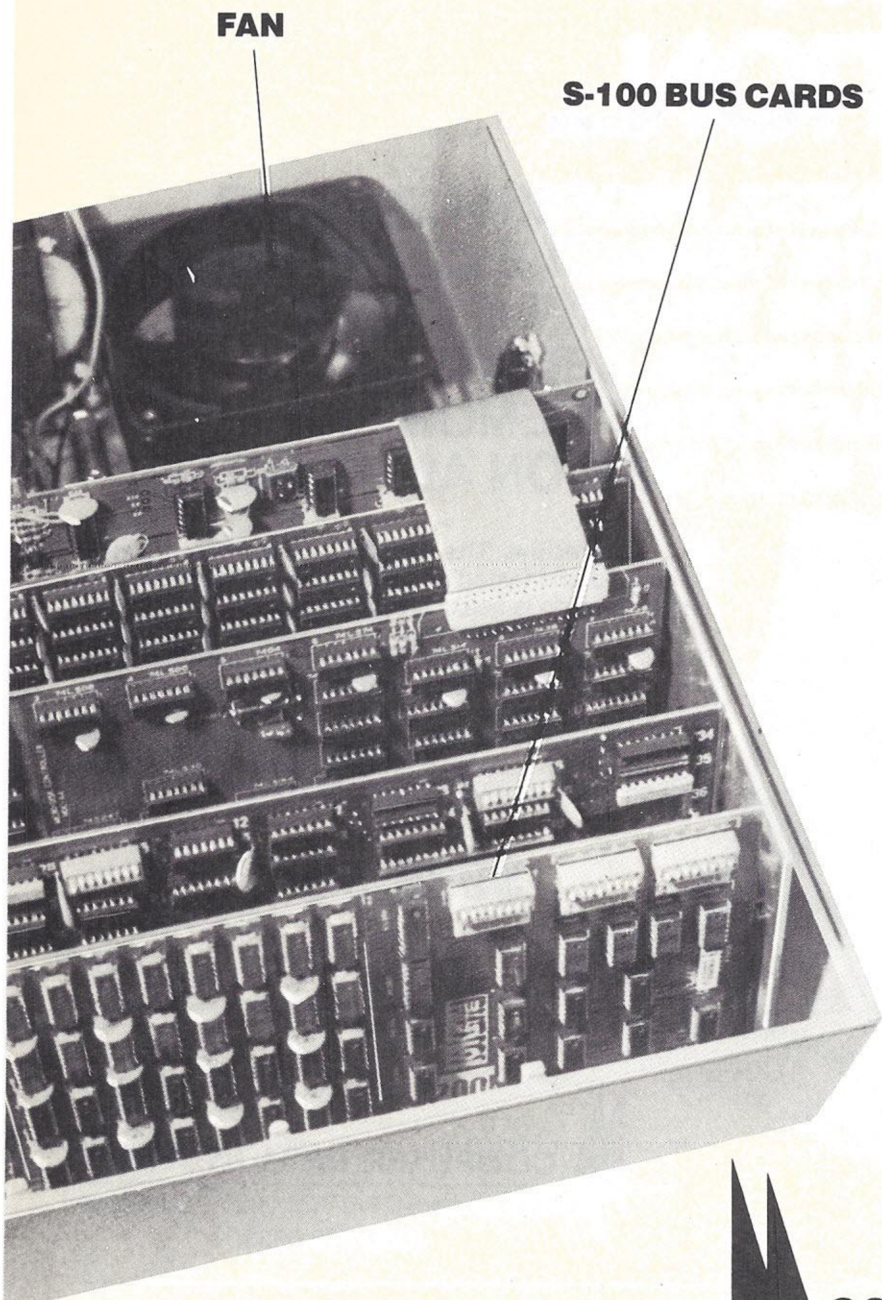


VERSATILE CRT®



VERSATILE II®

S IT?



SPECIFICATIONS VERSATILE II
Z-80 CPU
Video display on 9″ 64x16 screen
16K Static RAM Memory
Serial and Parallel I/O Ports with standard RS-232 Connector provided at rear of unit
Software to drive RS-232

EXPANDABLE
Add memory, printer, and up to 3 external mini-floppy disk drives.

OPTIONS
8K Memory Boards at $195.00 each.

SOFTWARE LIBRARY

Five diskettes are included to give you immediate programming capabilities.

DISK #1 contains a Disk Operating System and 12K Extended BASIC. Easy to use statements include: IF THEN, GOTO, READ, EXIT, FOR, NEXT. You get complete line editor, multi-statement lines and multi-dimension arrays. A BASIC teaching program on this diskette will have you quickly programming in BASIC.

DISK #2 has many games including STARTREK, BLACKJACK and STAR WARS. There's room left over for you to add your own.

DISK #3 is a Home Accounting Package with programs like Budgeting, and Checkbook Balancing.

DISK #4 contains a Small Business Accounting Package. Included are programs for Payroll, Inventory Control, Accounts Payable and Receivable, Taxes, Invoicing, Check Printing, and much more.

DISK #5 is a formatted blank diskette for you to enter your own programs.

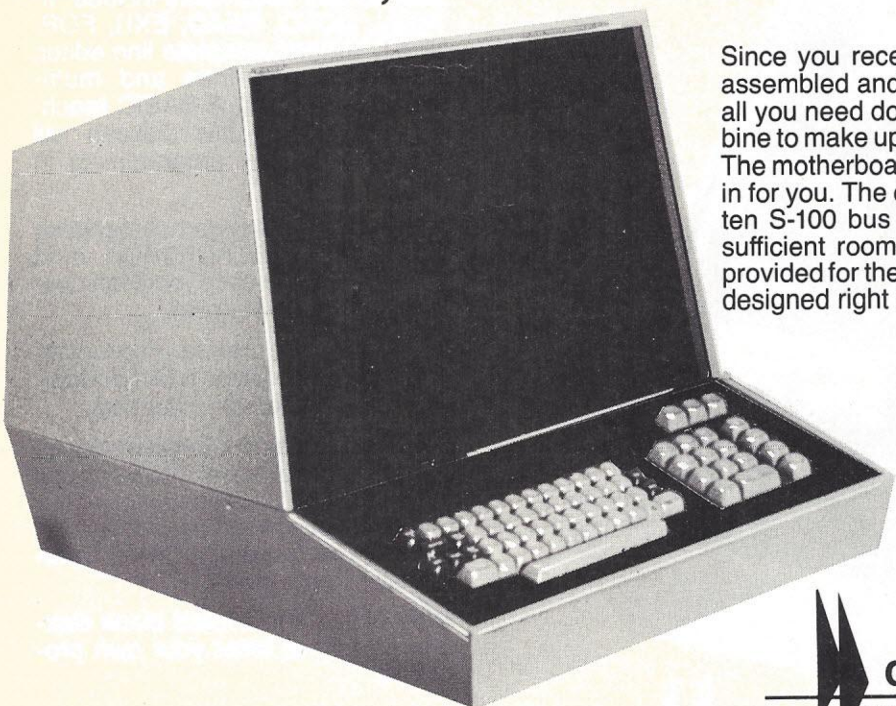
$2,495.00

COMPUTER DATA SYSTEMS
5460 Fairmont Drive Wilmington, Delaware 19808

CIRCLE 11

THE VERSATILE CRT® PROVIDES LIMITLESS POSSIBILITIES FOR YOUR OWN APPLICATION...

WHETHER IT'S A BUSINESS, PROFESSIONAL OR HOME APPLICATION, THE VERSATILE CRT® FILLS ALL THE REQUIREMENTS. THE VERSATILE CRT® IS THE ANSWER TO ALL YOUR NEEDS. SELECT AN APPROPRIATE CPU, SUFFICIENT MEMORY AND PERIPHERALS, AND IT WILL TAKE ON ALL COMERS.



Since you receive **The VERSATILE CRT®** completely assembled and tested and with full 6 months warranty, all you need do is select the components that will combine to make up the computer best suited to your needs. The motherboard has ten 100-pin sockets, pre-soldered in for you. The card rack provides space for as many as ten S-100 bus cards. The back panel has more than sufficient room for system expansion sockets. We've provided for the addition of a floppy disk drive with space designed right into **The VERSATILE CRT.®**

$999.95

COMPUTER DATA SYSTEMS
5460 Fairmont Drive Wilmington, Del 19808
(302) 738-0933

DISTRIBUTORS

Scientific Sales, Inc.
175 W. Wieuca, Suite 210
Atlanta, GA 30342
(404) 252-6808

Alexander and Company, Inc.
5518 Florin Road
Sacramento, CA 95823
(916) 422-9070

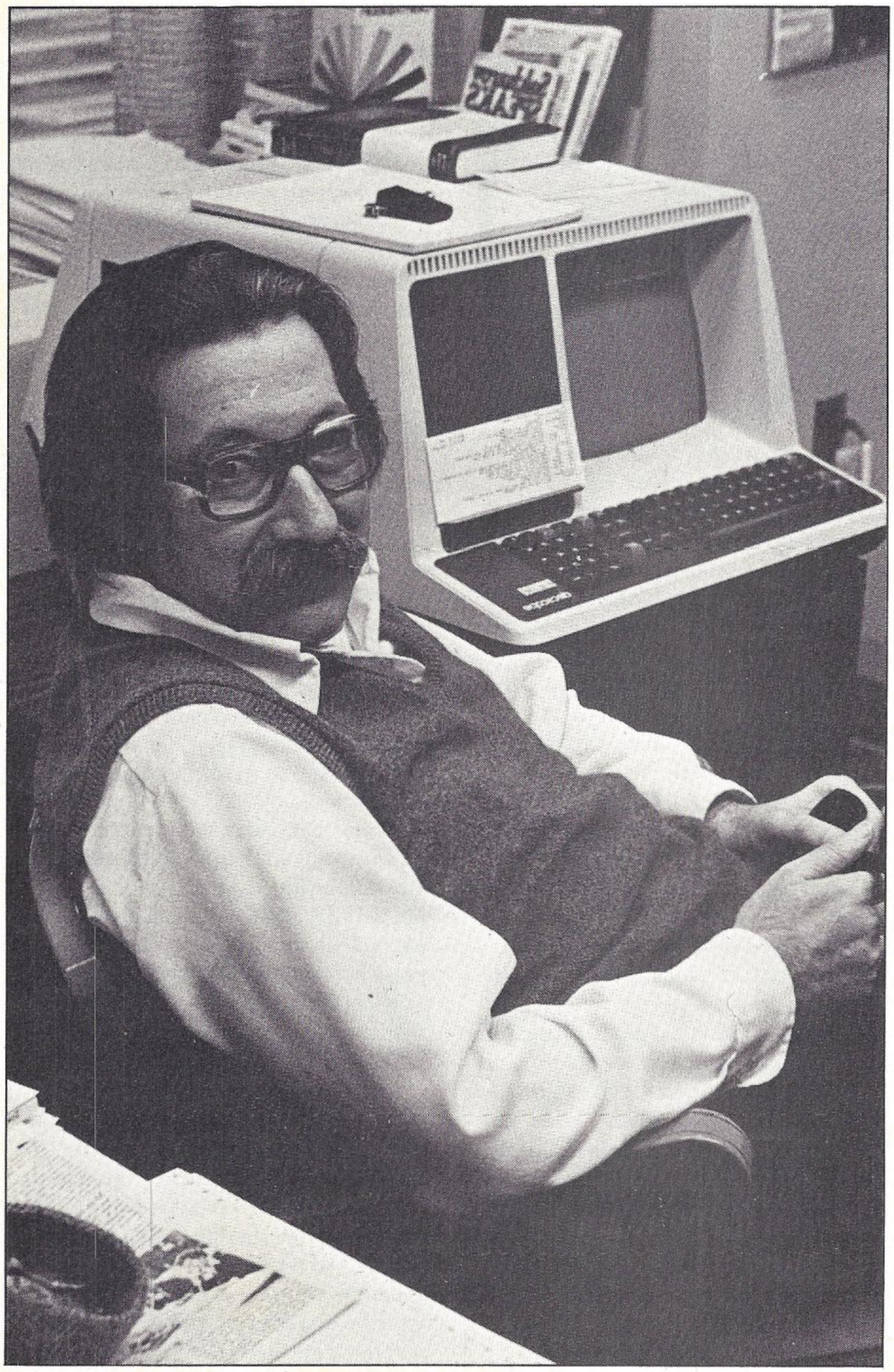
THE MICRO FORCE
Toll Free
1-800-824-5136 in U.S.
Ext. A-2090
P.O. Box 31265
Raleigh, N.C. 27612
1-919-781-3370

CIRCLE 12

Professor Joseph Weizenbaum:

Moral Considerations Of Artificial Intelligence

BY ERIC BENDER



Photographs by Eric Bender

"There are certain tasks which computers ought not do even though they can . . ."

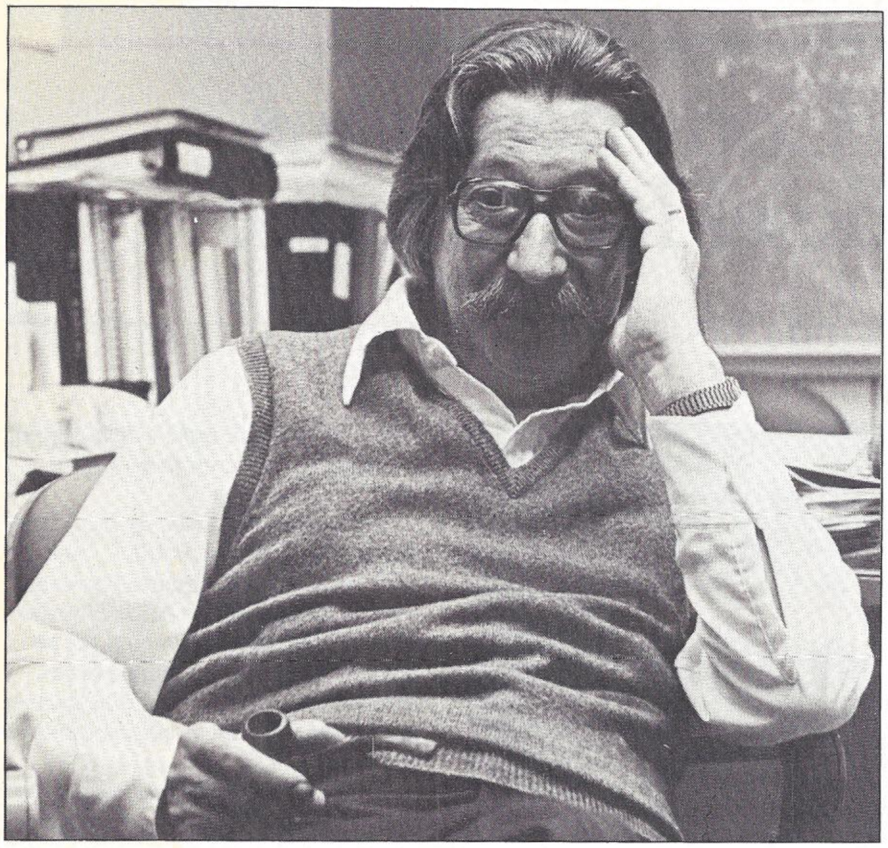
This statement, made by Professor Joseph Weizenbaum of M.I.T., summarizes his book **Computer Power and Human Reason,** *which has sold more than 23,000 copies, attracted the attention of not only computer scientists but also the general public and irritated the artificial intelligence research community.*

He wrote the book because of reactions to his earlier computer program ELIZA. In one ELIZA sequence, called DOCTOR, the computer interacts with the human operator like psychologist with patient. Weizenbaum said he was horrified when people took the interactions seriously (his secretary ordered him out of the room while she was talking with the DOCTOR). Some scientists even suggested that Weizenbaum's program might help to heal actual patients, a suggestion that disturbed Weizenbaum and prompted him to write the book.

Besides Eliza and the work on his book, he has taught at Stanford University, the Technical University of Berlin, Columbia University and Harvard. Some of his critics have jokingly taunted him by remarking, "My God! Can't the man hold onto a job?"

Personal Computing: In your book you described the computer addict – the "hacker" – as a programmer who couldn't stay away from the computer. Do you think personal computing hobbyists are liable to the same obsession?

Dr. Weizenbaum: Well, that's an interesting question. Let me show you an article that appeared in the October 29 *New York Times.* The author interviewed a senior programmer who says that he can't leave a program alone –



that he keeps fiddling with it and keeps thinking about it on the way to work.

This phenomenon, being caught up in programming until it becomes nearly an obsession, is widespread. As I tried to say in the book, becoming obsessed is a matter of degree. To a certain extent, all programmers suffer from getting "hooked" on computers. Programming seems to provide a special fascination, and a special gratification that comes partly from instant feedback.

Tell the computer to do something and sure enough it does something. It may do what you told it to do, or what you think you told it to do. Or it may do something else, which "criticizes" you and puts you in a kind of contest relationship.

Perhaps home computer hobbyists might also get hooked on computers; again, it will be a matter of degree. I doubt that the kids playing with daddy's computer will get hooked like this; I think it'll be kids with their *own* computers. In fact, we may have a new generation gap here.

Children may get very good at playing with computers. Their understanding of computers and their involvement may completely exceed that of their elders – particularly their parents.

Many years ago, someone would build an elaborate model railroad at home. That would be followed by the proverbial contest between father and son as to who would play with it. (You had the *Saturday Evening Post* cover with the kid sitting and waiting for his father to finish playing with the toy.) I think the exact opposite is going to happen with personal computers – the father won't want to play with the toy.

Today, when a child comes home and asks his father about a high school physics problem, he can't understand why his father doesn't know anything about it. In the future, the gap will be much, much larger.

The four-function calculator that parents use to balance the checking account is a much simpler machine than the multi-function calculator which the child brings home. That calculator costs a little more, but it has sines, cosines and other functions.

Very quickly now, cheap hand calculators will possess more and more storage capacity, and become moderately programmable – perhaps ten to a hundred steps. The barrier between what the child wants to do and what the parents understand will grow.

The calculator I've been talking about is the beginning of the home computer. You can physically modify the computer by taking out modules or adding them. You can also attach it to many home appliances – the TV set, maybe the electric typewriter, the sewing machine or the microwave oven. I think children are about to leave their parents far behind.

PC: Do you think parents will encourage this by saying, "Don't bother me; go play with your computer?"

JW: Television provides that kind of babysitter, unfortunately. Parents often abandon responsibilities to their children by simply plunking the kids in front of a TV set.

Computers might get the same job. Parents might say, "Don't bother me; go play Space War."

My hunch is that this approach won't work. Kids play with computers only if they're motivated. They get instant gratification only if a taste for it develops. To some extent, this taste is a function of the user's imagination, and a limited imagination leads to a limited attention span.

I don't think that the home computer will play the "idiot box" role that television does. It simply isn't that passive. It demands more.

PC: From your telephone conversation, I assume that your daughter at home was at a computer console.

JW: Yes. I've had a home computer for about 14 years, connected to M.I.T. computers. My children grew up thinking of the console as part of the furniture. I've permitted them to play with it, and I've encouraged them to some extent. They've never displayed great interest in it.

PC: Fourteen years ago you started the ELIZA program. How did that line of research begin?

JW: ELIZA was first published in 1966. I worked on it for a couple of years before that. It's an old program now, and it's been far superseded by Winograd's work and others in the natural language and computers field.

At the time, I had just come to M.I.T. to work on the CTSS program, which was the original timesharing system being developed under the direction of Professor Corbato. I think the history of timesharing is well known. People had noticed that large computers were idle for long periods while cards were being read or tapes were being written, and so on. During that time, someone else could use the computer.

Timesharing made computers more economical by using them more intensively. It also allowed, once again, a mode of computer interaction that had disappeared after the earliest days: hands-on, directly coupled, "conversational" computing.

Timesharing called for this kind of programming. It really wasn't necessary anymore to write a large FORTRAN program, fire up the computer to see if it worked, and then if it didn't work withdraw it and crawl over the output to see where the mistake was. All those steps one took in batch processing weren't needed now.

Rather, we could sit at the console and converse with the computer. It occurred to me that we could converse not only about program "bugs" but many other things as well – if we understood natural language, and small conversational fragments in specific contexts. For example, one types in, "How do I mount a computer tape?" That's a message and a context that a computer might "expect." But a message like, "When did Hannibal cross the Alps?" might not be expected.

It seemed to me that understanding these messages, decoding them and getting the computer to respond to them might be relatively straightforward. One requirement was recognition of certain patterns in conversational fragments. For example, "How do I do this operation on this computer?" The "do this operation" contains the meat of the question, but the "How do I" provides recognition of what's being asked.

That's how the program started. Working on my program-matching schemes, I quickly realized that if I wanted to test them in this mode I'd have to give the computer a very large data base. I wanted to avoid that problem. So I wondered what I could ask a computer which didn't require that the computer have too much knowledge?

Well, I got the peculiar idea that a psychiatrist might answer questions by phrasing in a way that would seem stupid in anyone else. When a psychiatrist asks a silly question, you think that he isn't being silly at all. He's just trying to draw you out.

That's how ELIZA started. It certainly had nothing to do with psychiatry, as I pointed out in the first article I wrote on it. I was puzzled, later, when people hailed it as the dawn of automatic psychiatry.

Salty reactions from concerned scientists

In the interview, Dr. Weizenbaum comments on the heated controversy his book raised. The lengthy controversy, for the most part, appeared as an exchange of letters in the newsletter of SIGART (Special Interest Group On Artificial Intelligence). The controversy raged through five consecutive issues. Here are some condensed excerpts (more can be found in the SIGART issues June 1976 to April 1977):

I had strong reactions to Joe Weizenbaum's book. The book mentions important concerns which are obscured by harsh and sometimes shrill accusations against the Artificial Intelligence research community. The personal attacks mislead the reader from more valuable abstract points.

– BENJAMIN KUIPERS, MIT AI lab (June, 1976)

JW says, "The very asking of the question, 'What does a psychiatrist know that we cannot tell a computer?' is a monstrous obscenity." On the contrary, it is a fantastically interesting and important question, deserving the attention of serious thinkers. What seems to worry JW is not the question but its potential for a foolish answer.

This moralistic and incoherent book uses computer science and technology as an illustration to support the view that science has led to an immoral view of man. I am frightened by its arguments that certain research should not be done if it might result in an obscene picture of the world. Worse yet, the book's notion of obscenity is vague enough to admit arbitrary interpretations by activist bureaucrats.

Weizenbaum's style involves making extreme statements which are later qualified by contradictory statements.

He says that science is not the sole or even main source of reliable knowledge. However, Weizenbaum doesn't propose any other source of knowledge or say what the limit of scientific knowledge is except to characterize certain thoughts as "obscene."

There are tasks Weizenbaum thinks computers should not be programmed to do – mostly for New Left reasons. One may quarrel with his politics, and I do, but obviously computers shouldn't do what shouldn't be done.

The objection to computers doing psychiatric interviews also has a component beyond the conviction that they would necessarily do it badly. Thus he says, "What can the psychiatrist's image of his patient be when he sees himself as an information processor following rules, etc.?" This seems like the Renaissance-era religious objections to dissecting the human body that came up when science revived.

– JOHN McCARTHY, Stanford University's AI lab (June, 1976)

Rebuttal: There is no correct or wrong view of life to which science or anything else can lead. The point McCarthy here misconstrues is that science, or any other system of thought, leads to an impoverished view of the world and of life when it or any system is taken to be the only legitimate perspective on the world and on life.

Whatever the merit of John McCarthy's review may be, it is all but undone by his assertion that the positions taken in the book are derived from a New Left idealogy. Not long ago the terms "pinko" or "commie" served the functions McCarthy assigns to the New Left in his review. I would have thought that people might have learned something from the

events of the tragic decades the United States has just passed through and that all participants in scholarly debate had by now renounced arguments by irrevelant political association.

McCarthy disagreed with the statement that Science is not the sole or even the main source of knowledge. How reliable would McCarthy say is his knowledge that his children are biologically his children or that the person he knows as his father is his biological father? Is science his source of such knowledge? Have we had reliable knowledge only since the founding of the Stanford Artificial Intelligence lab?

McCarthy is right in observing that tasks that should not be done at all should not be done by computers either. I am opposed to machine-administered psychotherapy and McCarthy cannot see what objections there might be to it if it were to "cure" people. I believe that machine-administered psychotherapy would induce an image of what it means to be human. The patient must never be lead into a situation where he is encouraged to regard himself as a mere object. My fear is that machine-administered psychotherapy necessarily induces this kind of self-image in the patient.
– JOSEPH WEIZENBAUM (Laboratory for Computer Science, Cambridge)

Rise and fall of reviewers' tidings

"This is the best book I have read on the impact of computers on society and on technology and man's image of himself. It is well written and not very technical. It is a book of reason, of humanity and wisdom."
– Keith Oatley, University of Sussex, in PSYCHOLOGY TODAY

"Judged as a philosophical work one must commend the effort rather than the achievement. The fundamental questions have been tackled better elsewhere. But then the traditional discussions may be ignored by the technologists on the grounds of irrelevance whereas they cannot dismiss Weizenbaum's account without coming to terms with his ideas."
– Seamus Hegarty, LONDON TIMES

"In a time when Weizenbaum feels so many of his colleagues are providing a science of answer finding, the professor instead lends himself to a science of question finding. This may prove to be the professor's most poignant contribution.
– Dr. Stanley R. Jaffin in SSC BOOKNEWS

"The book's final conclusions are frankly disappointing; that computers should not be used as a substitute for interpersonal human contact, nor should systems be introduced which have "irreversible and not entirely foreseeable side effects." Strict adherence to these rules would exclude virtually all known computer applications! But despite the lack of practically useful calculations, we should be grateful to Professor Weizenbaum who has said things humanists have been shouting for ages."
– Keith Haarhoff in NEW SCIENTIST

"Weizenbaum's polemic is brilliant and deserves the widest reading audience. The weakness of his theme is that he fails to make clear what kind of "social order" he is talking about: a democratic social order, totalitarian, post-industrial social orders?"
– Arnold Beichman, political scientist at U Mass Boston

"This rather sophisticated book is interesting and well-organized; unfortunately, its language is sexist."
– Richard J. Peppin in SCIENCE APPLICATIONS

PC: In your book you discuss that issue at length. What's the current status of automatic psychiatry?
JW: I don't follow it very closely, but apparently no one is trying to use this kind of program in actual therapy. I hope not.

Dr. Ken Colby at U.C.L.A. made the original suggestion that the program could help actual patients – handling hundreds of patients at state hospitals simultaneously. Dr. Colby has quit working on it. In a recent exchange of comments in print with me, he implied that the technical problems turned out to be much harder than he had thought. I infer that he still believes the idea is sound but the technical problems must be solved first.

I think, however, that this way of conversing with a patient fundamentally constitutes a deception. There is deception involved every time the computer says it "understands." That's wrong. The computer doesn't understand. Suppose the patient says that his wife left him and he's so angry he can hardly eat. In what sense can a computer possibly understand?

There's deception involved, and in my view psychotherapy should not be founded on deception.
PC: What is your current work here?
JW: I teach, as everyone here does. I'm involved in computer research, especially in secondary education. I don't want to say much about it because I think one should discuss results, not work in progress. But current education in computing topics leaves much to be desired.

One reason may be that academic workers in computer science have been diffident about working on apparently simple problems in secondary education. Clearly, we like to work at the most glamorous frontiers. However, some notable exceptions illustrate that simple problems in education are not simple at all. They are difficult and demand as much insight and scientific maturity as more glamorous problems.

In a way, technology is easy. If we know what we want to do, we can probably do it. The hard problems in education research are in teaching, not computer techniques.

As for my other work, I'm still deeply engaged in the aftermath of controversies raised by the book.
PC: Do your students read *Computer Power?*
JW: I don't know. I'm embarrassed to

ask. The book has been read widely, and it has caused considerable feedback. In many schools, the book's been adopted as a text. I'm particularly pleased that it's read for a variety of subjects, including psychology and philosophy. I get a fair number of letters from professors, and ordinary people, often agreeing almost entirely with me. I also get letters from people who react angrily.

PC: Most reviewers reacted favorably. Computer scientists seemed to have more mixed feelings.

JW: I think many computer scientists reviewed the book favorably. Most reviews in regular computer magazines have been generous, which was a special pleasure for me.

PC: Perhaps other scientists reacted to your description of a typical compulsive programmer, one who builds programs like enormous houses of cards.

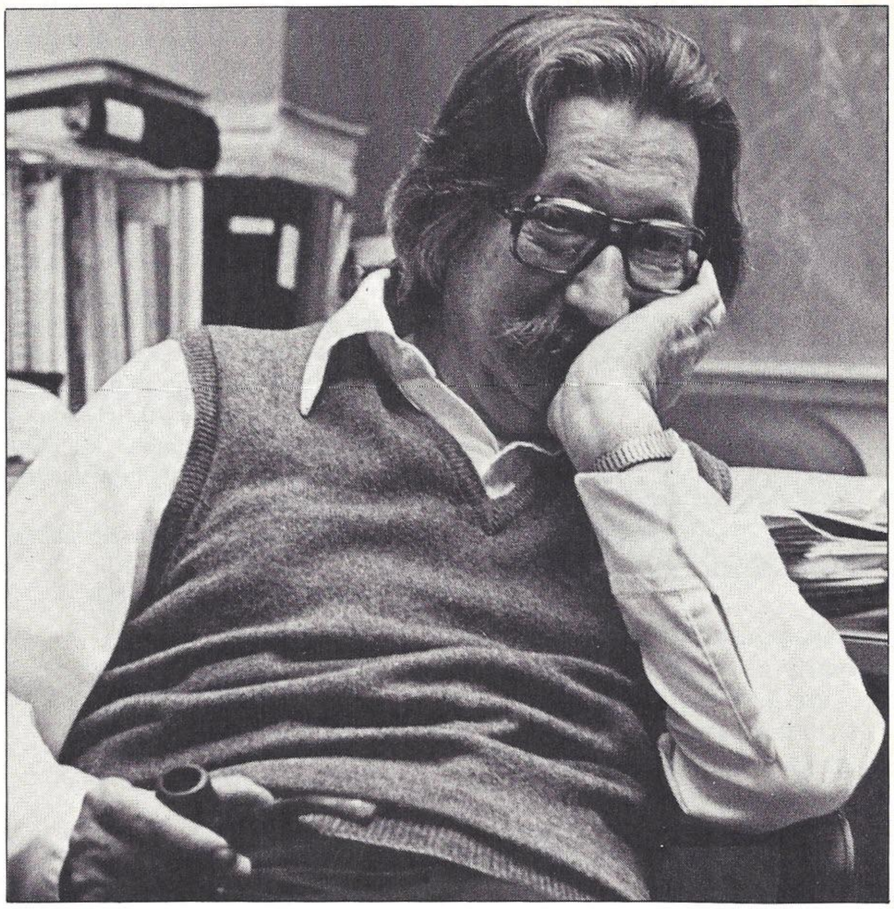
JW: Most programs are exactly like that. They run on no theory at all. The program is just made to work. If the program misbehaves in one little domain, you write another subprogram which detects errors and patches it up. But you don't know the side effects of that patch. Something else which worked well before might now misbehave. Most large programs running in the world today are exactly like that.

It's quite common for someone to write a program that few others, perhaps no one, can understand. In the *Times* article, the programmer is quoted as saying that six months after he writes a program, if he goes back to it six months later he doesn't know how he did it. Well, this means among other things that he doesn't understand the program.

Let me call your attention to that article again. It cites the program which runs typesetting for the *Times.* Now, when one looks at the article one finds serious typesetting errors. Any computer programmer can look at them and tell you what kind of programming error this represents.

The programmer here says that programs are either all right or all wrong. The article itself, which I think is hilarious, contradicts that, because the typesetting job isn't entirely wrong.

PC: When you point out this kind of error, does it help to clear the air? What effects did the controversies on your book have on the artificial intelligence research community?



JW: I flatter myself that speaking up as I did in the book has had some effect that at least on the surface looks positive. I notice that at the last International Conference on Artificial Intelligence, which was held at M.I.T. in August, many of the statements were more carefully qualified than they might have been a few years ago.

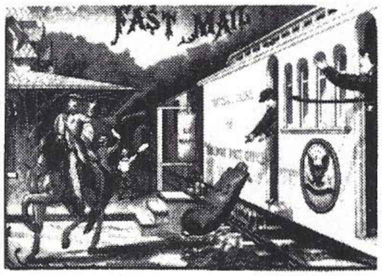
I'm somewhat cheered by that, but not very much. I suspect that the development is superficial. Fundamental beliefs in the ultimate computability of every aspect of life – beliefs that I feel are terribly mistaken – have not been seriously affected. But the controversy will continue.

Become a psychotherapist

In 1966 at MIT, Joseph Weizenbaum programmed the computer to imitate a psychotherapist. Here's a conversation between the computer and a patient, who complains that *Men are all alike.*

Men are all alike.
In what way?
They're always bugging us about something or other.
Can you think of a specific example?
Well, my boyfriend made me come here.
Your boyfriend made you come here.
He says I'm depressed much of the time.
I am sorry to hear you are depressed.
It's true. I am unhappy.
Do you think coming here will help you not to be unhappy?
I need some help, that much seems certain.
What would it mean to you if you got some help?
Perhaps I could learn to get along with my mother.
Tell me more about your family.
My mother takes care of me.
Who else in your family takes care of you?
My father.

Via Pony Express?



Have you written a letter, recently or submitted an aritcle – and have not heard from us? Please try again. Particularly if you wrote to our old Albuquerque office.

We've moved, as you probably know. Our new address is 1050 Commonwealth Ave., Boston, Mass. 02215. All your correspondence should be coming to this address. Much of the recent correspondence sent to the Albuquerque office has failed to turn up here in Boston.

When mail must be forwarded, it often goes astray, wanders around the nation's post offices like a lost dog. Eventually it will come home – when it gets good and hungry. Until then, we apologize for any delays in our communications. Perhaps a new computerized mail system programmed in BASIC, would speed things up?!

So, if you've written, or submitted an article or program, and haven't heard from us, please write again or call. We'll do our best to get back to you as soon as possible. Thanks for waiting.

The Editors

Become a Psychotherapist (Continued)

Your father.
You are like my father in some ways.
What resemblances do you see?
You are not very aggressive, but I think you don't want me to notice that.
What makes you think I am not very aggressive?
You don't argue with me.
Why do you think I don't argue with you?
You are afraid of me.
Does it please you to believe I am afraid of you?
My father is afraid of everybody.
What else comes to mind when you think of your father?
Bullies.
Does that have anything to do with the fact that your boyfriend made you come here?

Like the famous psychotherapist Carl Rogers, the computer asks probing questions that make the patient open up, but never gives advice: the patient must discover the truth for herself.

Instead of typing a single sentence, the patient can type a whole paragraph if she wishes. At the end of the sentence or paragraph, when she wants the computer to respond, she presses the carriage return twice.

I'll explain exactly how the program works, so you can become a psychotherapist yourself.

The computer begins by replacing some of the patient's words:

WORD THE PATIENT TYPED		REPLACEMENT	
mom	*you*	*mother*	*I*
dad	*my*	*father*	*your*
dont	*your*	*don't*	*my*
cant	*myself*	*can't*	*yourself*
wont	*yourself*	*won't*	*myself*
dreamed	*I'm*	*dreamt*	*you're*
dreams	*you're*	*dream*	*I'm*
I	*am*	*you@*	*are@*
me	*were*	*you*	*was*

For example, the sentence *Well, my boyfriend made me come here* becomes *Well, your boyfriend made you come here.*

The computer hunts through the revised sentence or paragraph for one of these *keywords:*

CATEGORY 8: *computer, computers, machine, machines*
CATEGORY 7: *name*
CATEGORY 6: *alike, like, same*
CATEGORY 5: *remember*
CATEGORY 4: *dreamt*
CATEGORY 3: *dream, if*
CATEGORY 2: *everybody, everyone, nobody, no one, was, your*
CATEGORY 1: *always*
CATEGORY 0: *are, are@, because, can, certainly, deutsch, espanol, francais, hello, how, I, I'm, italiano, maybe, my, no, perhaps, sorry, what, when, why, yes, you@, you're*

If it finds several, it chooses the one in the highest category; if they lie in the same category, it chooses the one the patient typed first.

Then it looks up the keyword in this phrasebook:

ALIKE, SAME In what way?
ALWAYS Can you think of a specific example?
ARE *are I . . .:* Why are you interested in whether I am. . . or not? *are. . .:* Did you think they might not be . . .?
ARE@ *are you . . .:* Do you believe you are. . .? *are:* Why do you say "am?"
BECAUSE Is that the real reason?
CAN *can I . . .:* You believe I can . . ., don't you? *can you . . .:* Whether or not you can . . . depends on you more than on me.
CERTAINLY, YES You seem quite positive.
COMPUTER, COMPUTERS, MACHINE, MACHINES Do computers worry you?
DEUTSCH, ESPANOL, FRANCAIS, ITALIANO I am sorry, I speak only English.
DREAM What does that dream suggest to you?
DREAMT *you dreamt . . .:* Really, . . .?
EVERYBODY, EVERYONE, NOBODY, NOONE Really, (everybody, everyone, nobody, noone)?

HELLO How do you do – please state your problem.
HOW, WHAT, WHEN Why do you ask?
I *I remind you of:* (see *alike*) *I are. . .:* What makes you think I am . . .? *I . . .you:* Why do you think I . . . you? *I . . .:* We were discussing you – not me.
IF . . . Do you think it's likely that . . .?
I'M (treat as *I are)*
LIKE *(am, is, are, was) . . . like:* (see *alike*) *like:* (get a different keyword)
MAYBE, PERHAPS You don't seem quite certain.
MY . . . Why are you concerned over my . . .?
NAME I am not interested in names.
NO Are you saying "no" just to be negative?
REMEMBER *you remember . . .:* Do you often think of . . .? *do I remember . . .:* Did you think I would forget . . .? *remember:* (get a different keyword)
SORRY Please don't apologize.
WAS *was you . . .:* What if you were . . .? *you was:* Were you really? *was I . . .:* Would you like to believe I was . . .? *was:* (get a different keyword)
WHY *why don't I . . .:* Do you believe I don't . . .? *Why can't you . . .:* Do you think you should be able to . . .?
YOU@ *you (want, need) . . .:* What would it mean to you if you got . . .? *you are . . . (sad, unhappy, depressed, sick):* I am sorry to hear you are (sad, unhappy, depressed, sick). *you are . . . (happy, elated, glad, better):* How have I helped you to be (happy, elated, glad, better)? *you was:* (see *was*) *you (feel, think, believe, wish) you . . .:* Do you really think so? *you (feel, think, believe, wish) . . . I:* (see *I*) *you are . . .:* Is it because you are . . . that you came to me? *you (can't, cannot) . . .:* How do you know you can't . . .? *you don't . . .:* Don't you really . . .? *you feel:* Tell me more about such feelings. *you . . .I:* Perhaps in your fantasy we . . . each other.
YOUR *your . . . (mother, father, sister, brother, wife, children):* Tell me more about your family. *your . . .:* Your . . . .
YOU'RE (treat as *you@ are)*

For example, if the keyword is *sorry,* the computer looks up *sorry* in the phrasebook, which says to print "Please don't apologize."

Suppose the patient types *If the job is lousy, he'll die.* The keyword is *if.* In the phrasebook, *if* is followed by three dots, which stand for the part of the clause that comes after *if,* which is *the job is lousy.* (The computer figures out where the clause ends by looking at the punctuation.) The phrasebook says to print "Do you think it's likely that the job is lousy?"

The symbol @ serves just to locate the correct keyword in the phrasebook. Thereafter, it is ignored.

Here's what happens if the keyword is *you@.* After locating *you@* in the phrasebook, the computer ignores the @. If the patient's revised sentence contains *you want . . .* or *you need . . .,* the computer prints "What would it mean to you if you got . . .?" If the patient's sentence contains *you are . . .sad,* the computer prints "I am sorry to hear you are sad."

For each phrase in the phrasebook, Weizenbaum stored a list of alternatives. Here are the alternatives to "Please don't apologize":

Apologies are not necessary.
What feelings do you have when you apologize?
I've told you that apologies are not required.

During a conversation with a patient, the computer keeps track of which phrases it has already printed, and uses the alternatives to avoid repetition.

If what the patient types contains no keyword, the computer may print one of the following:

I am not sure I understand you fully.
Please go on.
What does that suggest to you?
Do you feel strongly about discussing such things?

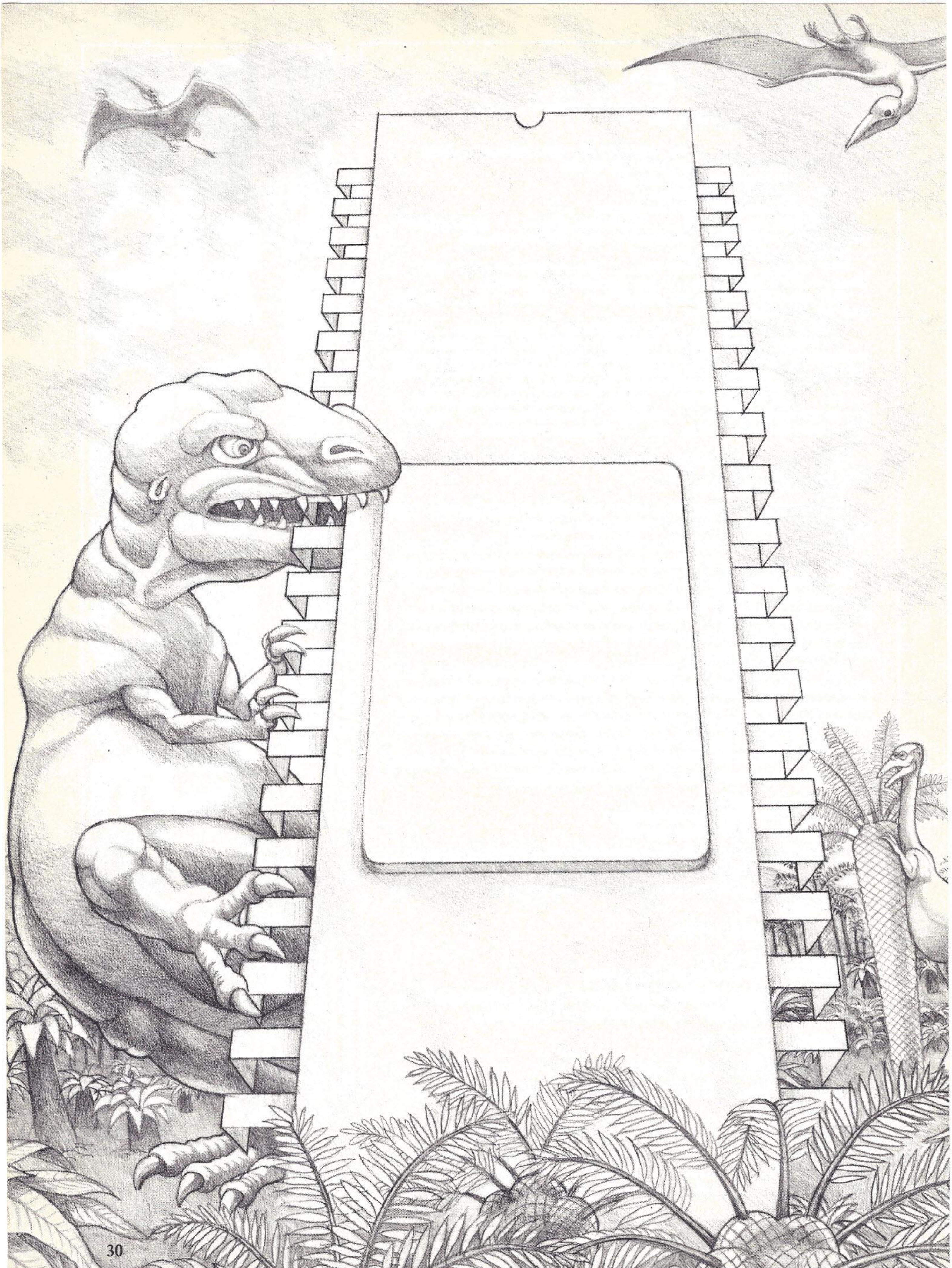
Or it may take a second look at earlier parts of the conversation, retrieve a clause that contained *your . . .,* and print one of the following:

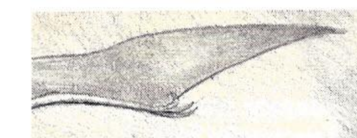
Let's discuss further why your . . .
Earlier you said your . . .
Does that have anything to do with the fact that your . . .?

For example, it may retrieve the clause *Your boyfriend made you come here* and print:

Does that have anything to do with the fact that your boyfriend made you come here?

The explanation above was taken from *The Secret Guide to Computers.* For the complete guide, send $11.50 to Russ Walter, 92 St. Botolph St., Boston, MA 02116.





MICRO HISTORY

BY RODNEY ZAKS

The emergence of microprocessors has been accidental. The first ones were often rejects of technology. Their initial design errors and inadequacies have been propagated until today, in the name of compatibility.

The emergence of working transistors can be traced back to the late 40s, after the war. Only 10 years later, the first working model for an integrated circuit was developed by St. Clair Kilby, from Texas Instruments. Around 1961, the first IC's were beginning to be produced in quantity.

The integration of circuitry then proceeded rapidly. SSI (Small-Scale Integration) appeared in 1964. It was a complete gate in a single chip. A gate is, for example, a "logical AND", "OR", "NOT"; it requires several transistors. In 1968, MSI (Medium-Scale Integration) appeared: it was a complete register on a single chip.

Commercial LSI (Large-Scale Integration) appeared in 1971: the first 1K-bit memory, UART and the first microprocessor. The first "general-purpose" microprocessor was introduced in late 1971 (Intel 4004).

The search toward higher integration resulted initially from NASA (aerospace contracts) and other government military contracts. In these programs the goal was miniaturization, *not* low cost. In the early 1970's research contracts from NASA and the government diminished rapidly, so the manufacturers looked for other sources of funds. Also at that time manufacturers of integrated circuits found for the first time an outlet for their products in the general public: the early desk calculators, followed by the pocket calculators. For the first time, it was possible to sell LSI to the general public.

In early 1971, only two standardized LSI products had been developed: the first 1K-bit dynamic RAM and the first UART's (Universal-Asynchronous-Receiver-Transmitter – a serial-parallel converter). Nobody could imagine then what the next standard product could be.

The first microprocessor

The design of the Intel 4004 microprocessor, introduced in 1971, resulted from a contract with a Japanese desk calculator manufacturer. In fact, the early purchasers of the 4004 (a 4-bit microprocessor) had to sign a contract committing themselves not to develop a desk calculator with the chip for at least one year. This first general-purpose microprocessor was in fact, designed as a calculator. It was, therefore, not powerful, and generally inadequate for general-purpose computing. .

The next significant event was Intel's introduction of the 8008, the first general-purpose 8-bit microprocessor, in 1972. A few years before, Display Terminals Corporation, now known as Datapoint, had generated a request for bids for the production of a monolithic processor that could control a CRT in a single chip. Two companies vied for and obtained the development contract: Texas Instruments and Intel. After months of effort, Texas Instruments withdrew. Intel continued the development and came up with a component which could essentially satisfy all of the functional requirements of Datapoint except one: it was too slow. In addition, at about the same time, a price war had flared over the prices of bipolar components, resulting in significantly lower prices.

Because of the inadequate speed of the processor developed by Intel, and of the much lower prices of bipolar components, Datapoint decided to implement its controller in bipolar technology. Intel was left with a chip whose development had been paid for, but for which there was no obvious market. An essential product of Intel, then a very new company, was memory products. The 8008 was therefore (reluctantly?) introduced on the market, with the assumption that it would sell more memory chips. All design efforts seemed to be halted and the design team assigned to other tasks. That was to be the end of microprocessors!

Today's microprocessors

To the presumed surprise of the manufacturer (and of its competitors), sales of the new microprocessor started increasing rapidly. Intel realized quickly the potential of this new device, reassembled a design team, and introduced a year later the successor of the 8008, the 8080. Simultaneously, all the leading competitors set to implement their version of what an 8080 should have been, had it been designed correctly from the start. Within the two years that followed, all of the leading "standard" microprocessors now on the market were to be introduced, most of them inspired by the early design of the 8080. Motorola introduced the 6800 (nearly one year after the 8080), Rockwell introduced the PPS8, Signetics the 2650, and so on.

We have now entered a third generation of microprocessor design, where the successors to the 8080 and the 6800 have arrived on the market. They are the Z80 from Zilog, the 8085 from Intel, and the new 1-chip microcomputers (the F8 from Fairchild and Mostek, the 8048 from Intel, the PPS4 from Rockwell, and the TMS 1000 and 9940 from Texas Instruments).

Silicon Valley

The LSI industry was born from two development poles in the U.S.: Bell Telephone Labs (in New Jersey) and the region around Sunnyvale (south of San Francisco) now known as "Silicon Valley". Many, if not most, of the companies populating Silicon Valley are born from Fairchild.

Shockley (the Nobel Prize physicist) had assembled a small team of scientists on the East Coast about 15 years ago. Within this team were Robert Noyce and Gordon Moore. Shockley left Bell Telephone in order to create his own company, Shockley Research Laboratories. A few years later, a small group of scientists left his company and created a new one, Fairchild, in what was to become Silicon Valley. In 1968, a small group of scientists left Fairchild and created still another company in Sunnyvale, "Integrated Electronics" (Intel). Intel was founded by Robert Noyce and Gordon Moore and introduced three years later the first "microprocessor". Two years ago, three of the leading designers of the 8080 left Intel to create their own company, Zilog, which introduced a successor to the 8080, the Z80. History does not yet say whether some leading designers of Zilog products intend to separate soon from this company to create their own Silicon Valley. This mechanism of company creation and implantation has been typical of the electronics field, and in particular, of the integrated circuit field. The leading microprocessors on the market today have strikingly similar features (and design inadequacies).

A closer look at Intel

The ancestor of 8-bit microprocessors, the Intel 8008, introduced in 1972-1973, was not intended to be a general-purpose microprocessor. It was to be a CRT-display controller for Datapoint. For all its design inadequacies, and its limited performance, the 8008 was an overwhelming success and even motivated a number of leading semiconductor manufacturers into producing competing designs.

The 8080 was designed as a successor to the 8008. To maintain compatibility with the 8008, the 8080 included all the 8008 registers plus more, and all the 8008 instructions plus more. The 8080 was the first powerful microprocessor introduced on the market and still is the best-seller in its class.

Several other microprocessors of similar performance were introduced on the market a year or more later. Technically, the 8080 is no longer the best product on the market. But the 8080 still has had the highest sales of any standard microprocessor, because it was introduced first, and probably because Intel was among the first companies to invest in the developing support chips and support software for its product. All of the 8080 competitors were introduced after a nine month delay (Motorola 6800) or more. The 8080 has now a "successor" product, the 8085.

Motorola

Motorola introduced the 6800 in direct competition to the 8080. The 6800 has essentially the same internal architecture as the 8080. It has some differences at the register level. The 6800 is equipped with two accumulators, versus one for the 8080, but it has a small number of general-purpose registers. The 6800 has a special IX (index) register, which facilitates access to tables stored in the memory. The 8080 does not have an index register, but is equipped with register-pairs which can be used to provide a similar facility. The 6800 instructions probably reflect the fact that it was introduced after the 8080: they tend to be somewhat more complex but generally similar to those of the 8080. Overall, either of the two microprocessors comes out marginally faster, depending on the function used in the comparison. The most significant difference in performance is achieved, not by comparing a standard 8080 to a standard 6800 (their performance is essentially similar), but by simply purchasing a faster version of either the 8080 or the 6800. The 8080 is available in three versions, the standard 8080A with a 2MHz clock, the 8080A-2 and the 8080A-1 with a 3MHz clock. The 6800 is now available in two versions also.

The standard 6800 uses a 1 MHz clock. Should this mean that the standard 8080A is twice as fast as the standard 6800? No. The clock simply supplies the pulses needed by the internal microprogram of the control unit. The 8080, on the

The best-selling microprocessors have been copied by a variety of manufacturers, both for profit and to acquire experience in technology and processor design.

average, requires twice as many micro instructions as the 6800 does, so their overall performance is similar. A "typical" 8080 instruction is executed in two microseconds, and most 6800 instructions are executed in two microseconds.

A useful comparison would be to execute a "standard program" on both microprocessors, but there is no such thing as a standard program. When comparing the 8080 and the 6800, an 8-bit by 8-bit multiplication may be faster on one of the processors, while an 8-bit by 16-bit multiplication may be faster on the other, due to "features" (in fact, design errors or design peculiarities).

Second-sources

Within the industry, there are two kinds of second-sources for a microprocessor: authorized second-sources and non-authorized second-sources. The result is the same: another manufacturer produces a component which should be identical to the model. This could cut into the sales of the first component, but it provides the market stability most industrial and military purchasers require and usually contributes to disseminating of the chip. Mutual cross-licensing has become fashionable these days. The best-selling microprocessors have been copied by a variety of manufacturers, both for profit and to acquire experience in technology and processor design. Non-authorized copies are often also called "scanning-electron microscope" versions.

For example, the 8080 has only one operational authorized second-source so far (even though other companies have agreements authorizing them to produce the 8080 if needed): it is Siemens in Germany. This manufacturer has the masks and the process to manufacture the chip, and does. But the majority of the 8080 second-sources are non-authorized. They are, for example: the AMD9080 form AMD (a fully compatible version of the 8080 claimed to be somewhat more reliable), the 8080 from Texas Instruments, from NEC (in Japan), and from National Semiconductor. Most of these components are claimed to be totally compatible with the 8080, both pin-for-pin compatible and software compatible. Some of the early versions were not completely compatible and resulted in surprises to the users.

The impact of second-sources on the market can be very significant. AMD can probably be credited with starting the major price war on microprocessors. It resulted in a price drop on the 8080 and similar products which was one of the most spectacular events of 1975. At a time when Intel was selling the 8080 for $70 (always in quantities of 100 plus), AMD announced the 9080 for $35. Within days, Intel was announcing the 8080 at $35. The 9080 from AMD is under $12. Other manufacturers, and in particular Texas Instruments, have merrily entered the price war. As a result, it is widely suspected that the manufacturers of microprocessors which do not sell in significant numbers are in fact losing money with every chip.

Most other microprocessors also have their second-sources. The 6800, for example, is second-sourced by American Microsystems (AMI) in the U.S.A. and by the Sescosem Division of Thomson CSF in Europe.

National Semiconductor

National Semiconductor introduced the SC/MP as its first 8-bit design. SC/MP stands for "simple cost-effective microprocessor". It was intended as a very simple microprocessor (few extra chips required), very inexpensive. It is indeed a simplified microprocessor and it is indeed cheap. But a low

cost for the MPU is not sufficient to generate significant sales. The P-channel version of the SC/MP was slow. A new version is now available in N-channel technology, which has improved speed. This microprocessor is well-suited for some simple applications but does not have the processing power of an 8080, a 6800 or a 2650.

In 1976, National introduced another 8-bit microprocessor: its own 8080 implementation (compatible with Intel).

MOS Technology

MOS Technology has introduced a variety of microprocessors on the market. They are the 650X "family". The term is misleading: there are important differences between the various family members. One of the first family members, the 6502, has strong analogies with the 6800 and may be considered as a competitor. Its bus organization, internal registers, and instruction set are very similar to the 6800. In fact, support devices for the 6500 and 6800 are interchangeable. The 6502 is faster than the 6800, and the special components introduced by MOS Technology (such as combination of memory and I/O) result in systems built with less chips, for smaller or medium-sized systems. They are also priced very low, resulting in inexpensive small- to medium-sized systems with a processing power similar to, or greater than, the 6800.

Intersil

The Intersil 6100 is a 12-bit CMOS microprocessor that emulates (imitates) Digital Equipment Corporation's PDP-8. The PDP-8 has been the most successful minicomputer in the world. The Intersil 6100 is the only 12-bit microprocessor on the market (excluding the Japanese Toshiba TLCS12 developed for Ford Motor Company). It executes all the instructions of the PDP-8. Because it is made of CMOS, a complete Intersil system can be powered with ordinary batteries. The software compatibility with the PDP-8 lets you program in PDP-8 instructions. But you can use ROM chips directly with this microprocessor the way you would with a standard microprocessor. Because the PDP-8 stores a subroutine's return address at the subroutine's beginning, subroutines cannot be stored in ROM, unless you "adapt" the programs in order to store the return addresses into a separate RAM.

Zilog

Three of the main designers of the 8080 left Intel and created their own company, Zilog, in Los Altos. Zilog is financed by Exxon, the oil company. Its first and essential design was the Z-80, which incorporates in a single chip the 8080, its 8224 clock, and its 8228 system controller plus additional facilities. It is completely compatible with 8080 software, is as fast as the fastest 8080 version, and the new Z-80A operates more quickly than the fastest 8080. The Z-80 has several other advantages over the 8080.

Intel again

Intel could not remain idle while facing the threat presented by the Z-80. It was obvious that with the progress of technology, the extra chips required by the 8080 could be integrated in a single chip. The 8085 is Intel's answer. The 8085 does incorporate the 8224 clock, the 8228 system controller and the 8080 within a single chip. It does operate at the speed of the fastest 8080. But it does not offer some of the features of the Z-80.

Reprinted from Microprocessors: from Chips to Systems, *by Rodney Zaks. Copyright ©1977 Sybex. Entire book ($9.95) from Sybex Inc., 2161 Shattuck Ave., Berkeley, CA*

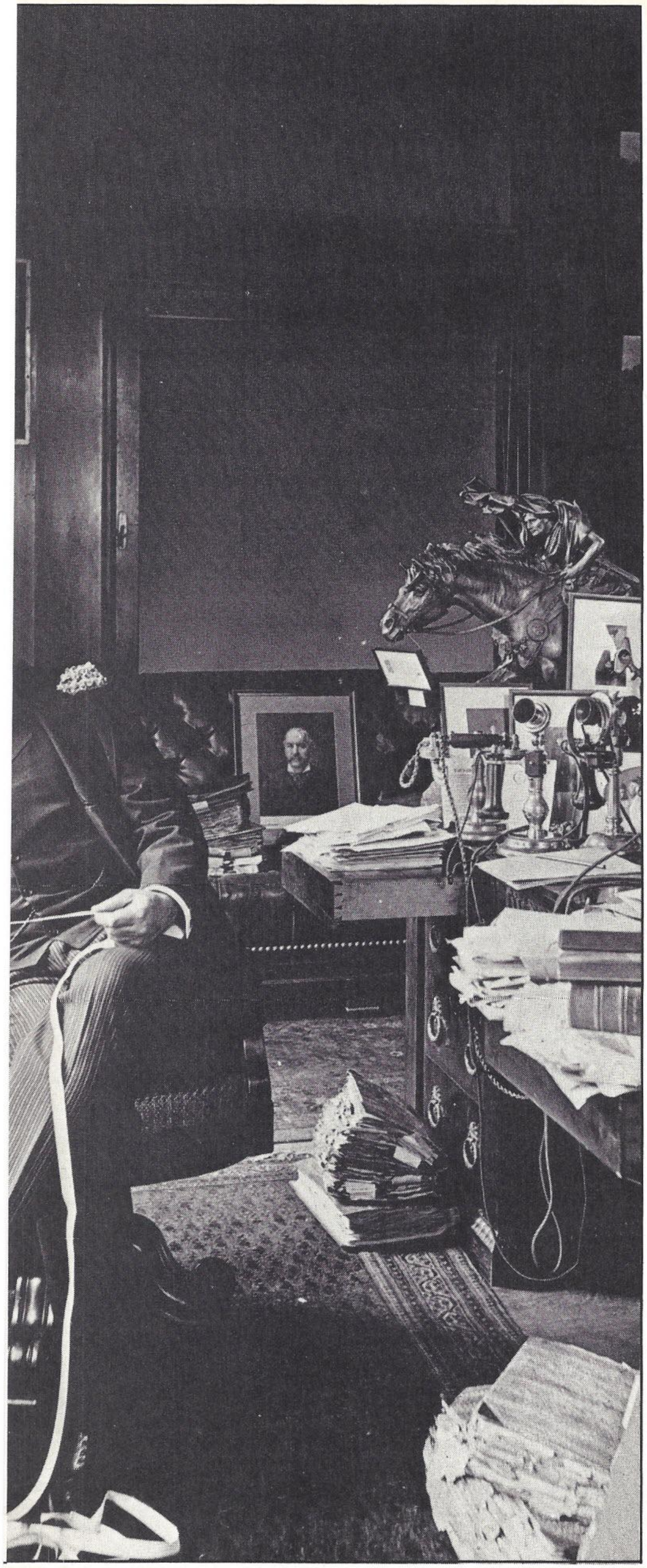
A Plunge into the Stock Market

BY TOM MUNNECKE

The Bettmann Archive, Inc.

Want to cash in on the personal computing bonanza? Tired of turning your garage into a mail order factory? Tired of slaving over mysterious hexadecimal codes in your own software shop? Maybe you're just plain lazy. Perhaps you don't want to risk your time and money. Cheer up, boys and girls. There's a place for you. It's called the stock market.

Suppose you decided that the new Commodore Pet was going to be a winner when you first heard some rumors last March. If you had called your broker, he probably would have said he didn't like Commodore International (CBU) even at the bargain price of 4½. In fact, some brokerage houses were recommending that the stock be dumped. The company was not doing too well with its calculator business and had just written off other losing enterprises.



That didn't bother you, you old plunger. You knew things were going to get bullish in the market – take off. So you invested your little nest egg of $10,000 for 4300 shares at 4½. You held on all spring and watched the stock climb merrily to 14½ in late June. Your little nest egg was now worth $62,000. Commodore turned out to be the hottest growth stock on the American exchange that quarter.

Suppose you decided not to sell at the top, waited a few weeks then sold out at 13½. You would have then realized a $45,000 profit on your $10,000 investment in 4 months. Not bad, but what do you do for an encore?

Enter Tandy Corporation and its new Radio Shack TRS-80 computer. Tandy was in a turnaround situation in August when it unveiled its new computer. They had taken a bath on CB's by overstocking themselves with 23-channel sets just when the FCC approved 40-channel units. The oversupply was finally corrected, however, and they were looking forward to a good Christmas quarter. Many analysts began recommending buy Tandy.

Tandy stock was selling at about $26 at the time, down from its year's high of $42. But instead of buying the stock, you decided to buy "options." Options gave you the right to purchase stock at a fixed price only until a pre-agreed expiration date. For that right, you paid only a fraction of its purchase price. However, if the stock did not reach the sell price the option would become worthless. If the price of the stock rose the percentage gain of the option's price would far exceed the percentage gain of the underlying stock's price. Therefore, in general, option price exaggerates price movement of the stock.

Suppose you bullishly reinvested your $55,000 bundle in Tandy options when Tandy announced its TRS-80 model in early August. At that time, options to buy Tandy-stock-at-$30-by-April-1978 were selling for 1¾. So you plunged in with your dough and snatched up 30,000 of them. You waited a few weeks for the news to hit the media, watched the stock climb from 26 to 31 and you sold your options for a cool 4½ a share. Your nest egg was now worth a nice $130,000. In retrospect, anybody can turn a $10,000 investment into $130,000 in 6 months. But how do you determine what will happen in the future? What will make a company's stock take off?

The interesting thing to note in these two true examples is that the stock moved completely on the *expectation* of

As a concerned computerist, you should be able to ferret out which products might make a significant contribution to the field, and thereby increase its market success.

the product. Commodore's stock took off 6 months before a single PET was delivered. Radio Shack waited much longer before announcing their product, but its sales would not show up on the balance sheets until the middle of next year.

Commodore's stock kept pace with the public's awareness of what was happening. Each new article in a national publication boosted Commodore up a point or two. The Commodore rise was more spectacular than Tandy's because Commodore's price was lower to begin with, and the entire company's success rested on the project.

So how do you find out what is going to happen? This author found that conferences are a good place to start. The Commodore PET prototype was first shown to the public at the last San Francisco Computer Faire. Mixing with the spectators surrounding the first PET were all of Commodore's top brass, principals of most of the other hobby computer companies, and magazine reporters. The reporters would shortly go home, write their articles, and wait 3-8 weeks for them to hit the press. If you were there. . .

Obviously, you cannot attend every show in the country. But you can listen to the local grapevine at the computer store or computer club. In addition, computer magazines, such as *Personal Computing, Electronic Engineering Times,* and *Electronic News,* can also provide timely insights into events before they appear in *Newsweek.*

As a concerned computerist, you should be able to ferret out which products might make a significant contribution to the field, and thereby increase its market success.

Street talk . . .

Whenever you discuss the stock market, you'll always hear reference to street talk. What are they saying on the street? Although the reference is usually to Wall Street, it might just as easily refer to the sidewalk in front of your local stock broker. In this area, then, if you keep your ears open, you can pick up all sorts of interesting information. But how you use it is another matter. The wise eavesdropper consults his broker. The plunger treads where wise men fear to go.

Whether or not you invest should depend on what you have spotted in the prospectus of the company. This prospectus, or stock description, is something anyone can obtain free of charge before investing. Or you might go to a business library and dig out the information yourself, from Standard and Poor (S&P) reports. (S&P is the bible of the stock market.) You can always find traders pouring over updated S&P reports, trying to spot something there that might make the stock a good buy.

The following "street talk" might be worth considering. Company names have purposely been omitted since the Securities Exchange Commission frowns on anyone "pushing" a particular stock. Supply your own. It really doesn't change matters much.

"The __________ Company has grown 15% in sales in the past 3 years. It's worth buying."

"__________ has just applied for a patent for a new type of rectifier. The patent is good for 17 years and if it goes over, the __________ will dominate the market. Sounds like a good gamble."

"__________ Inc. has just enlarged its research and development staff. Wow! Any company that will do that has something up its sleeve. Take a chance."

"__________ and __________ are trying to raise capital. Well, they're not going to build a new plant with my money. No sir!"

"The __________ Company is a small outfit. Big brokerage houses aren't fooling around with small companies. This company, though, has dynamic ideas. They're small and they're going places. Small companies usually take off like rockets. Big companies simply trudge steadily along satisfied with their usual profits. Put a few chips on the small fellow. He can use your money and he could make you a millionaire."

"__________ Inc. is paying a big dividend this year. That means they have no place to put their money. No new ideas. No new products. No expansions. Nothing. Sales staying constant. They've simply stopped growing. So, stick with a smaller company. They're the growers."

"The __________ Company has a growth rate of 4% annually. Tends to follow that, year after year. But it pays good dividends. Therefore, seems like a nice, dependable old lady to trust your money with."

"Most brokers say not to invest in __________ because it's reached a state of 'growth-arrest'. But it seems like a good time to pick up a bargain on the market – buy when others are selling. More millionaires have been made that way than any other way. (But many guys and gals have lost their shirts, too".)

"Good Heavens! 3 o'clock! Market is closed! Let's go have a drink someplace and see how much money we've made today."

Street talk fades away and becomes a mumble of feverish exclamations during happy hour in the stock market bar rooms. One thing about street talkers and street listeners – whatever it is they are talking about – it keeps them happy.

Not everything is that simple, however. Trading on Commodore stock was suspended twice last summer. Once, for a few hours due to excessive trading, which did not affect the price. In September, however, trading was suspended for a few days, after which it opened down 5 points from its previous 18 7/8. The question, and concern, was whether Commodore could actually produce the machine they had demonstrated.

What does the future hold in store for personal computing and the stock market? How many more Commodores and Tandys will come along before the public wearies of them? I believe the public is not really aware of them yet, and the market certainly has not reacted to any realized profits yet. And the really big fish – Texas Instruments, Hewlett-Packard, IBM, and others, have not even announced yet.

If one had to wager on a scenario, one would have to anticipate Texas Instruments coming into the market within the next 24 months. If they do, they would probalby operate as they have with the calculator and digital watch markets. They would undercut competitor's prices by about 30%, and offer a mature product with extensive literature, software and support. They would probably introduce the product through their consumer outlets. So the first you might know about it may be when you see one on the shelf at your local department store.

What should this do to the market? One would expect Texas Instrument stock to rise, as did Tandy and Commodore. But TI is a much larger company, and its stock is already high priced. Therefore, one couldn't expect too much of an upward trend. TI options are traded, so that should make it a little easier to take advantage of relatively small price increases.

But a major TI announcement would have a great effect on competitors – particularly Commodore. Here, now, is your opportunity to cash in on the personal computing market – selling short on stocks you anticipate will go down. Selling short is the process of selling stock you don't own, on the promise of buying it back at a price presumably less than you sold it for.

What about other companies, such as MITS, IMSAI, and Processor Technology? First of all, MITS was acquired by Pertec Computer Corporation, which has just recently moved to the New York Stock Exchange. Secondly, it does not appear that MITS is going to do all that well. In fact, some insiders suspect that Pertec "got a lemon" when they bought MITS. IMSAI and Processor Technology are not publicly traded, and may share the same fate as MITS. If the S-100 bus fades into obscurity, these companies may follow as the big boys enter the picture. As the old Chinese proverb says, "Big fish eat little fish, and little fish eat dirt."

But what will the big fish take over, and what will be left over for the little fish? Certainly the mainframe such as the PET can benefit from the economies of scale. The manufacturer can distribute the cost of software and operating system over a large number of units, thereby delivering a cheaper and more sophisticated product. But this will, in turn, create secondary markets to fill in product gaps, introduce innovative products, software, retailing and servicing. These markets are inherently small, entrepreneurial business – the little fish.

What is the moral of all this? You can participate in the personal computing market – without turning your garage into a factory. Happy hunting. And remember, buy low, sell high, and watch out for the big fish. PC

Smart Investing

After reading Tom Munnecke's article on investing we decided to ask a professional about this business. Donald Regal of Fahnestock & Co., nationally-known stockbrokers, sent along this report.

When considering an investment in the fast technology industry an investor must realize that what is new today may be old hat by tomorrow and thus his personal guidelines and investigations prior to a financial commitment are essential. A cautionary checklist is a must even for the individual who considers himself well informed and in the forefront of science and perhaps even sophisticated in the new technology in the state of art. A prudent investor's checklist should include some obvious points of reference.

The product

Is the product that has attracted our investor's attention a prototype? Is it the result of pilot production? Is the

"A wise man knows all the roads he travels."

A cautionary checklist is a must even for the individual who considers himself well informed and in the forefront of science . . .

product the start of a full scale production run? In short, knowing this answer can determine how soon the product will reach the marketplace in volume.

Is the product one that can be readily copied by a competitor? On the other hand, is the product protected by patents or perhaps a secret proprietary process?

The next step is to determine if something may be in the wings. Does a competitor have a well kept secret about an equal or superior product that might possibly be first or coincide with our new product in the market place? Is anyone currently conducting research that may well make our product obsolete in a relatively short time? In effect, we are trying to determine how long our "better mousetrap" will be better than the competition's. Once having developed the product, management must successfully market its new invention. Obviously such bad slogans as NO OTHER EQUIPMENT WILL DO WHAT THIS PRODUCT DOES – OUR PRODUCT DOES THINGS FASTER AND BETTER AND MORE ECONOMICALLY THAN ANY OTHER PRODUCT would be expected to be part of the marketing program. Regardless of catchy phrases and colorful brochures the product must still be produced economically and priced consistently to what customers are willing to pay.

Hand in hand with the above is the size of the marketplace. The two obvious questions are: How big is the market, and how large is the profit margin? Basically the determination may boil down to a dollar figure. Will it be a case where bottom line figures will make a big impact on a small company or a small impact on a big company?

The company

Our next step is to realistically assess the company in question. How good is management? Often, smaller companies are started by technical wizards and indeed they are essential. One must look a little deeper at the corporate structure to be sure that these technicians can manage, can market, can make a profit, or to determine that they have indeed employed such personnel.

Is the company adequately capitalized? Often small companies are not strong enough financially to finish the job they start. Production facilities are another area that should be investigated. What is conceived in a laboratory or a garage is a far cry from adequate production space and organized production line. Another key area of attention should be directed to distribution network. Building the product is one thing; maximizing distribution is a science in itself.

Stock pricing

Perhaps the most nebulous area of all is what price should one pay for the common stock of a corporation. Much has been written on this subject through the years and it is a sure bet much will continue to be written in the future. There is no one answer to the question and an intelligent investor should be aware of that fact. How high is high and how low is low are questions that can not be answered.

It would appear safe to say that in most instances the market price of common stock of a company should or is entitled to increase in actual earnings and perhaps future prospects of the company. Logical as the previous sentence may appear, all investors should realize that such is not always the case.

Timing is a key to profitable investing. The first thing an investor should do is make note of the current market price and its price in the immediate and intermediate past. By such an observation an investor may surmise that the price of the common stock has already run up a fat percentage. Perhaps there are others who know or suspect there is something favorable in the wings? Some investors may chose to hold back their purchase until the company itself gives publicity to its new product.

Other investors may chose to wait until the product reaches the market. Still others may chose to wait until actual profits come down to the bottom line.

Obviously it is the wish of all investors, their investment advisors and their stockbrokers to back the right company with the right product at the right time; however, this is not an easy task. Merely finding and identifying the most promising product and prospects is a difficult task. Many flowers are cultivated to bloom but, as is often the case, the bloom is short lived. To an investor this is often translated into capital losses.

Perhaps the above checklist may be helpful in eliminating companies you contemplated purchasing – as their foreseeable problems are evident. There is no substitute for homework in this area. Perhaps one consideration an investor should always keep in mind when he is searching for a potentially profitable investment in the stock market is that whatever one buys, someone is always taking the opposite position.

Bewitched, Bothered and Bewildered?

Selecting your first computer can be made easier if you keep a few basic ideas in mind.

BY WILLIAM MILLER

Choosing a computer these days can be as bewildering as it was to buy your first family car. The tires had to be kicked, the doors slammed, the horn tooted, the stick shift jiggled, and everyone had to take his turn sitting in the driver's seat before you could make your final decision. Once made, the entire family piled into the flivver for a ride around the block under the envious stares of the neighbors.

One great difference separates your first car experience from buying your first computer – when you bought your car you were able to drive it home from the showroom immediately; when you buy your first computer system, you'll first have to lug it home, set it up, and work out the bugs before it moves at all.

If you're vague about your objectives the options available to the first-time buyer can be especially confusing. But selecting your first machine can be made easier if you keep a few basic concepts in mind.

Four alternatives are open to the prospective computer buyer. Each offering should be studied carefully to decide which computer system best meets your needs.

Your alternatives include buying a complete "turn-key" system complete with computer hardware, system software and application software that performs specified tasks; buying a system with system software and hardware (forget about application software and simply adapt the system software to your use); purchasing only the hardware and developing both system software and application software; or developing your own hardware as well as both softwares.

When you don't have the resources or time to develop an application, or when the "fit" is near perfect, turnkey systems will suffice – although these systems do have some drawbacks you should be aware of. "Tailoring", for example, is usually required to assure fit. Most standard application programs require modification before you can put them in production. Some-

Sometimes the amount of tailoring required for a software package justifies starting from scratch.

With too great a cost and too many drawbacks to consider in the first alternative, you'd be better off selecting the second alternative. Because both system software and hardware are supplied you can directly develop the applications it lacks. The chief problem with following this route lay in delay and cost overrun if the system software cannot develop application software.

Few people choose the third alternative – hardware only. Most purchasers lack the specialized knowledge required to develop system software. But you might select this alternative when nothing else is available at a price you can afford.

The fourth system, where you develop both softwares and the hardware, should be chosen only when you require a highly specialized application with a minimum of hardware/software design and development. Don't even consider this system unless you have the expertise required to develop all the components.

After you've selected your system you must consider and resolve a number of questions. How easy is it to write a program? test a program? debug a program? change or modify a program? When the programs are fully tested how easily can you operate the system? Does operating the system require specially trained personnel? Are the programs or files protected from modification while in production? Can the system process work within production time limits? Can the system be expanded? What languages are supported? Because implementation of language differs from machine to machine, what features are supported? What are the capabilities and limitations of those languages? Are file structures flexible? Can you add peripherals (printers, disks, tapes) without rewriting software? Is there a command language? Can you execute from pre-stored command files? Can more than one person use the system? Are user files guarded against deletion or change by other users? Is the system easy to operate?

Costs

The cost of a system should include the price of system software or the cost of having it developed. Elements of such system software are comprised of operating system, programming languages, maintenance, utilities, command language, data access (I/O) and schemata, plus the cost of developing application software. When purchased with system hardware, system software will often cost less then "in house" development.

"Choosing a computer these days can be as bewildering as it was to buy your first family car."

Reliability

The time a system can be used productively is referred to as "up time". "Down time" is the time a system cannot be used because of software or hardware malfunction. Most down time during initial phasing-in of a system results from software difficulties. After systems have been debugged, software becomes less of a factor. At that point, hardware takes over as the most frequent cause of down time although the advent of solid state technology has reduced failure rates to low, acceptable levels.

Peripherals, more than other components, continue to be most vulnerable to failure. Again, the purchaser must decide what degree of reliability he needs. Professional use requires greater reliability than educational or hobbyist use.

You should be able to obtain service for all parts of your system (application software, system software and hardware). Most computer stores provide service on a contractual basis, but the increased reliability of solid state technology, is leading the trend away from maintenance contracts to fixed-rate service calls. Both options might be offered by a computer store.

Maintenance of microprocessor systems has already been reduced from electronic boards to identification and replacement of the faulty board. Mechanical components require constant maintenance; cleaning, oiling and adjusting. Consequently, mechanical components are more time consuming than solid state components.

Hardware

If the cost of hardware continues to decline as the cost of software rises, hardware will soon represent less than 5 percent of a system's total cost.

While distinctions were formerly made between microprocessors, minicomputers and computers, the rapid development of micros has relegated these distinctions to historical dictionaries. The most significant difference among microprocessors involves the number of bits they handle. 16-bit processors are faster and more flexible than 8-bit processors, while 32-bit processors tend to be faster and more flexible than 16-bit processors.

Factors to be considered when evaluating hardware include: Is it standard? Are there second sources for memory, interfaces, controllers and peripherals? Will the components readily adapt to a more advanced system? Are power supply and mother board capable of growth? Will the hardware be able to use newer technology as it becomes available?

Your evaluation of a computer system involves three items: application software, system software and hardware. But this evaluation should take place after you've developed a clear idea of your objectives. PC

Definitions for the neophyte

Software: specific sets of instructions which, when executed by a computer, perform predictable activities. Software is usually stored in magnetic media; such as, disks or tapes. Information may also be stored on non-magnetic paper tape, punched cards or coding sheets.

System Software: programs which are usually prepared by the manufacturers of hardware. These pre-written programs make development and production of application software practical. Included in this category are: operating system, compiler, interpreter, file editor, command language and various utilities such as a sort program.

Applications Software: a set of programs which, when executed by the computer, produces meaningful work. Such meaningful work might include payroll, general ledger, accounts receivable, inventory control and sales analysis. Application software may consist of a single program. Often, though, it contains many programs.

Hardware: tangible electronic and mechanical components which comprise a computer system. This system includes the computer itself plus memory, controllers, interfaces, power supplies, cables and peripherals (printers, disk drives and keyboards). Peripherals are also referred to as Input/Output (IO) devices.

A Mega·Success in 1977!

NOW PREPARING FOR ITS SECOND SHOWING

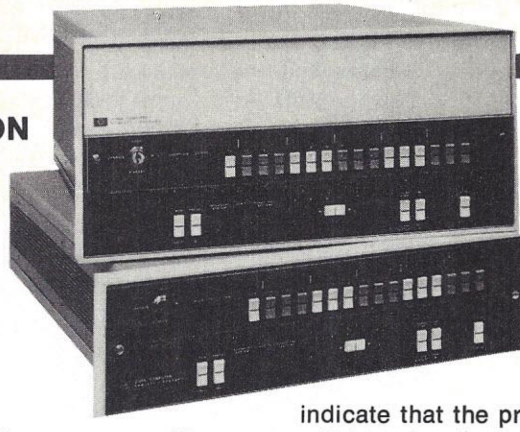
International Microcomputers Minicomputers Microprocessors '78

20, 21, 22 JUNE, 1978

Palais des Expositions Geneva, Switzerland

ANOTHER PRIME OPPORTUNITY TO SELL TO THE EUROPEAN MARKET

Microcomputers, Minicomputers, Microprocessors, Peripherals, Components and Services



A 50% LARGER EXHIBITION IS PLANNED

The sales results IMMM obtained for its 1977 exhibitors is clearly indicative that in 1978 more and more producers will be displaying products for use in every type of industrial, commercial, consumer and military application. Their enthusiasm has prompted many additional manufacturers of small computers (firms which attended and observed in 1977) to make serious commitments regarding participation in the next show. With these new exhibitors and the increased space already requested by 1977 participants, IMMM '78 will be a much larger show!

In 1978, the kind of people you want to meet — executives, engineers, designers, manufacturing and support supervisors, and others — will be out in force . . . to see, to learn, to BUY. And you will want to be there with YOUR products and services.

PROGRAMME DESIGNED TO ATTRACT MANY MORE VISITORS

The remarkably large audience of highly qualified and seriously interested visitors who attended the first IMMM exposition was obviously pleased with the technical programme. Comments indicate that the programme, as well as the exhibition, will be a key factor in attracting an even larger group of attendees to the next show.

The 1978 programme, chaired and presented by internationally recognised experts, again will be designed to offer the kind of practical solutions to day-to-day problems that attendees seek. A special session on "Tips for Hobby Microcomputers" is being planned.

SPONSORED BY DISTINGUISHED PUBLICATIONS

Mini-Micro Systems (U.S.A.) **Polyscope (Switzerland)**
Micomp (Switzerland) **Markt Und Technik (West Germany)**

For Additional Information:

IN EUROPE:
Mr. Bert Saunders
Kiver Communications S.A.
(U.K. Branch Office)
171/185 Ewell Road
Surbiton,
Surrey KT6 6AX England
Telephone: 01-390-0281
Telex: 929837

Mr. Ernest Jungmann
Promotion Marches Exterieurs
Residence Mexico
65, rue du Javelot
75645 Paris CEDEX 13,
France
Telephone: (1) 583-96-62
Telex: 210500F

IN THE UNITED STATES:
Mr. Joseph Maurer
Industrial & Scientific Conference Management, Incorporated
222 West Adams Street
Chicago, Illinois 60606
Telephone: (312) 263-4866
Telex: 256148

IN JAPAN:
Mr. K. Yamada
ISCM Japan
Kokado Building
1-3-18 Akasaka
Minato-ku, Tokyo 107,
Japan
Telephone: 03-585-8321
Telex: 28887

Big Memories for Micros

BY CHIP A. TYETI

If you're contemplating adding big memory chunks onto your microcomputer you may find all the options confusing. The accompanying chart gives you the facts on what's available in 16K or greater random access memory (RAM) – the differences in capability, flexibility, availability, price and more. If you're not ready to add on to your system yet, just tear out the chart (or photocopy it) for future reference.

In the months ahead, we hope to provide similar charts comparing small memories, PROM and ROM boards, CPU boards, I/O boards, floppy disk operating systems and other equipment needed to enhance your personal microcomputer system. We're even open to your ideas for additional comparative charts.

Random access memory, as you might know by now, is the memory which stores data by address at any random location you assign to it. You can retrieve it, pump it out to a cassette, floppy disk or TTY, modify it, execute it, write completely over it, or even (with appropriate software) move the data to some other location. In other words, RAM memory is the flexible storage space which you need in operating a micro system.

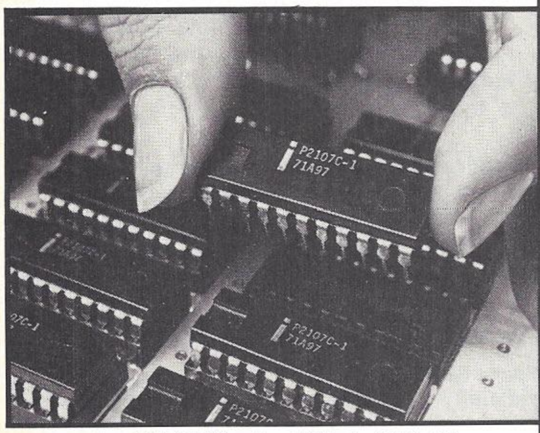
RAM differs from ROM or PROM or EPROM in that you can write into it immediately with no complex procedure or additional equipment. Without going into it in detail, let's say for now that ROM's, PROM's, and EPROM's, are essentially read only memories. That is, you cannot write into them at all – not even one byte.

Most micros keep a maximum of 64K or 65,636 bytes of direct storage on-line. Some memories (e.g. Cromemco) allow for more than 64K on line by using the "memory bank select" feature which provides for 8 banks of 64K bytes. An 8-position DIP switch on each board addresses any of the 8 banks and turns on and off at output port 40H. A "1" turns it on; a "0" turns the memory card (16K) off. This feature also allows for a time-sharing configuration, using a minimum of software. Eight people could utilize a maximum of 64K each, and only one bank of 64K will be on-line at one time.

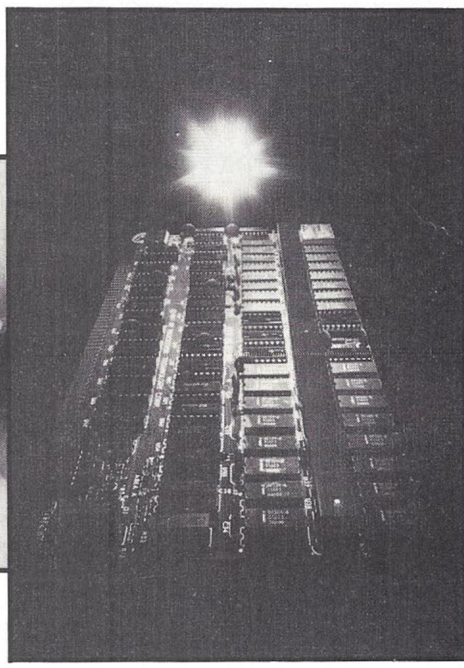
Some microcomputers have a limited number of slots available in the mother board or daughter board. The denser memory boards obviously take up fewer slots in totaling 64K, yet there may be no corresponding increase or decrease in the ratio of dollars per kilobyte. For example, four 8K RAM memories might cost $130, on the low end, in kit form. That's $520 for 32K or $16.25 per kilobyte. The high end might be $225 for 8K or $900 total; or about $28.13 per kilobyte.

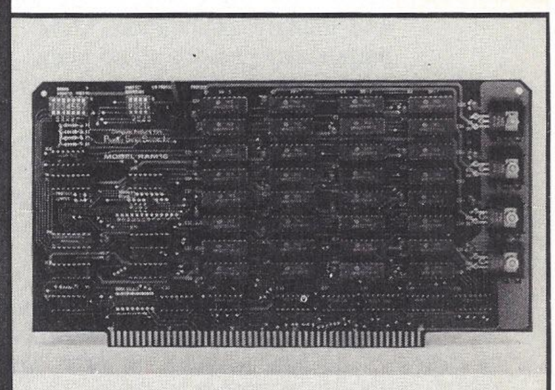
Comparing the low and high ends of a single 32K board, we see $259 and $795 respectively. Accordingly, that's about $8.09 per kilobyte and about $24.84 per kilobyte. Depending on your system's requirements, your piggy bank, board availability, your time and your aesthetics, the cost per kilobyte can vary over a relatively wide range.

Static vs. dynamic RAM boards should also be considered. Static boards generally require more power than dy-



Intel 2107C family of 22-pin 4K RAMs


Processor Technology 16KRA Semi-kit


Problem Solver RAM-65 16K Static Memory

Mail Order/ Retail Dealers	Warranty Period	Manufacturer	Size in K's	Price in $'s	Static/Dynamic	Upgradeable	Memory Access Time*	Memory Cycle Time*	Availability **	Memory Chip Used ***	Bus Structure	Memory Protect	Kit/Assembled	Parity Option	Wait States	Smallest Block Addressing	Power Consumption	Catalog Number
MO	120D	ADVANCED COMP. PROD.	32	$ 995	S	N	500	-	1	2114	S100	Y	K	N	0	32K	2.4w	S10032K
MO	120D		32	1095	S	N	500	-	1	or	S100	Y	A	N	0	32K	2.4w	S10032K
MO	120D		24	859	S	N	500	-	1	9135	S100	Y	K	N	0	24K	1.8w	S10024K
MO	120D		24	959	S	N	500	-	1	or	S100	Y	A	N	0	24K	1.8w	S10024K
MO	120D		16	565	S	N	500	-	1	4045	S100	Y	K	N	0	16K	1.2w	S10016K
MO	120D		16	665	S	N	500	-	1		S100	Y	A	N	0	16K	1.2w	S10016K
RE	Y	APPLE COMPUTER	16	600	S	N	500	-	1	any 16K	Apple	N	A	N	0	1 unit	18w	AZCO16X
MO	90D	ARTEC ELECTRONICS	16	545	S	Y	250	-	1	4045	S100	N	K	N	0	4K	9.75w	32K100-16
MO	90D		16	695	S	Y	250	-	1	4044	S100	N	A	Y	0	4K	9.75w	32K100-16
MO	90D		32	1055	S	N	250	-	1	4044	S100	N	K	N	0	4K	19.5w	32K100-32
MO	90D		32	1205	S	N	250	-	1	4044	S100	N	A	Y	0	4K	19.5w	32K100-32
MO	90D	BASE $_2$	16	325	S	N	250	-	1	4044	S100	Y	K	N	0-3	4K	8.5w	16KS-Z
MO	90D		16	285	S	N	450	-	1	4044	S100	Y	K	N	0-3	4K	8.5w	16KS-B
MO	Y	CENTRAL DATA	16	289	D	Y	450	-	1	2108-4	S100	N	A	N	0	16K	16w	16KRAM
MO	Y		32	525	D	N	450	-	1	2108-4	S100	N	A	N	0	16K	16w	32KRAM
RE	Y	CREA COMP SYSTEMS	16	560	D	Y	250	575	1	2108	S100	Y	A	Y	0	4K	.5w	M100/16
RE	Y		16	485	D	Y	250	575	1	2108	S100	Y	A	N	0	4K	.5w	M100/16
RE	Y		32	990	D	N	250	575	1	2116	S100	Y	A	Y	0	4K	.5w	M100/32
RE	Y		32	885	D	N	250	575	1	2116	S100	Y	A	N	0	4K	.5w	M100/32
RE	90D	CROMEMCO	16	495	D	N	200	-	1	4050	S100	N	K	N	0	16K	15.4w	16KZ-K
RE	90D		16	795	D	N	200	-	1	4050	S100	N	A	N	0	16K	15.4w	16KZ-W
RE	90D	CYBERCOM	16	449	S	D	275	400	1	410D	S100	Y	K	N	0	4K	2.75w	MB-7
RE	90D	DIGITAL GROUP	16	445	S	Y	450	450	1	4044	DG	N	K	N	0	16K	7.5w	MEM16
RE	90D		16	495	S	Y	450	450	1	4044	DG	N	A	N	0	16K	7.5w	MEM16
RE	90D		16	375	D	Y	450	-	3	4044	DG	N	A	N	0	16K	1w	MEM16
RE	90D		32	650	D	N	450	450	3	4044	DG	N	A	N	0	32K	1w	MEM32
RE	90D		32	750	D	N	400	400	1	4115	S100	N	A	N	0	4K	3.2w	32K
RE	90D		32	945	S	N	450	450	1	4801	DG	N	A	N	0	32K	15w	MEM32
RE	90D		64	1850	D	N	400	400	1	4116	S100	N	A	N	0	4K	3.2w	64KD
MO	30D	DIGITAL MICROSYSTEMS	16	525	S	D	300	300	1	2114	S100	Y	K	N	0-1	4K	10w	16KRAM
MO	Y		16	595	S	D	300	300	1	2114	S100	Y	A	N	0-1	4K	10w	16KRAM
RE	Y	DYNABYTE	16	555	S	N	250	250	2	4044	S100	Y	A	N	0	4K	9w	1625
RE	Y		32	995	S	N	250	250	2	4044	S100	N	A	N	0	4K	20w	3225
RE	Y		16	525	S	N	450	450	2	4044	S100	Y	A	N	0	4K	7w	1645
RE	Y		32	925	S	N	450	450	2	4044	S100	N	A	N	0	4K	15w	3245
RE	Y		16	399	D	N	250	500	1	5271	S100	N	A	N	0	16K	5w	16KDYN.
MO	90D	ELEC. CONTROL TECH'Y	16	450	S	N	450	450	1	4044	S100	N	K	N	0	16K	12w	16K RAM
MO	90D		16	506	S	N	450	450	1	4044	S100	N	A	N	0	16K	12w	16K RAM
RE	Y	EXTENSYS	16	595	D	Y	300	-	1	2109	S100	Y	A	N	0	8K	1w	RM64-16
RE	Y		32	895	D	Y	300	-	1	2109	S100	Y	A	N	0	8K	1w	RM64-32
RE	Y		48	1195	D	Y	300	-	1	2109	S100	Y	A	N	0	8K	1w	RM64-48
RE	Y		64	1495	D	N	300	-	1	2109	S100	Y	A	N	0	8K	1w	RM64-64

Legend

MO Mail Order only
RE Sells through Retailers
D (Warranty Period) prefixed by number of days
Y (Warranty Period) one year, usually assembled units
D (Upgradeable) smaller memory capacity available and incrementally upgradeable
Y (Upgradeable) upgradeable to larger memory capacity (using different chips, jumpers, etc.)
* Memory Access Time and Memory Cycle Time are in nanoseconds
** 1=within one month; 2=between one and two months; 3=beyond two months
*** Only the numeric code is listed in indicating the chip used. There should be little difficulty in determining the manufacturer from this code

Lasting Memory Listing

Mail Order/ Retail Dealers	Warranty Period	Manufacturer	Size in K's	Price in $'s	Static/Dynamic	Upgradeable	Memory Access Time *	Memory Cycle Time *	Availability **	Memory Chip Used ***	Bus Structure	Memory Protect	Kit/Assembled	Parity Option	Wait States	Smallest Block Addressing	Power Consumption	Catalog Number
RE		GIMIX	16		S		450			4044	SS-50					4K		
RE	Y	IMSAI	16	499	D	N	400	475	1	2104-A4	S100	Y	A	N	0	16K		RAM16
RE	Y		32	799	D	N	400	475	1	2108-3	S100	Y	A	N	0	16K		RAM32
RE	Y		64	2649	D	N	400	475	1	2116-3	S100	Y	A	N	0	16K		RAM64
RE	Y		16	449	D	N	400	475	1	2104-A4	S100	Y	K	N	0	16K		RAM16
RE	Y		32	749	D	N	400	475	1	2108-3	S100	Y	K	N	0	16K		RAM32
RE	Y		64	2599	D	N	400	475	1	2116-3	S100	Y	K	N	0	16K		RAM64
RE	90D	MITS	16	360	D	N	350	-		D2104A	S100	N	K	N	0	16K	2.5w	88-16MCD
RE	90D		16	395	D	N	350	-		D2104A	S100	N	A	N	0	16K	2.5w	88-16MCD
RE	90D		16	680	S	N	215	390		4200	S100	N	K	N	0	16K	4.9w	16MCS
RE	90D		16	785	S	N	215	390		4200	S100	N	A	N	0	16K	4.9w	16MCS
RE	90D	NORTH STAR	16	399	D	N	200	-	3	4027-3	S100	N	K	Y	0	8K	9w	16KRAM
RE	90D		16	459	D	N	200	-	3	4027-3	S100	N	A	Y	0	8K	9w	16KRAM
RE	60D	OHIO SCIENTIFIC	16	596	S	N			2	4200	OSI	N	A	N	0	16K		CN3
MO	90D	POLYMORPHIC SYSTEMS	16	495	D	N	375	525		4096	S100	N	A	N	0	4K	10.4w	005200
MO	Y	PRIME RADIX	32	870	D	Y	300	450	1	2108	S100+	N	A	N	0	8K	15w	32K
MO	Y		40	990	D	Y	300	450	1	2108	S100+	N	A	N	0	8K	15w	40K
MO	Y		48	1110	D	Y	300	450	1	2108	S100+	N	A	N	0	8K	15w	48K
MO	Y		56	1230	D	Y	300	450	1	2108	S100+	N	A	N	0	8K	15w	56K
MO	Y		64	1350	D	N	300	450	1	2108	S100+	N	A	N	0	8K	15w	64K
RE	120D	PROBLEM SOLVER SYS.	16	599	S	N	250	-	1	2114	S100	Y	A	N	0	4K	6.5w	RAM16
RE	120D		16	570	S	N	450	-	1	2114	S100	Y	A	N	0	4K	6.5w	RAM16B
RE	120D		16	599	S	N	250	-	1	2114	S100	Y	A	N	0-2	4K	6.3w	RAM65
RE	120D		16	599	S		450	-	1	2114	S100	Y	A	N	0-2	4K	6.3w	RAM65B
RE	90D	PROCESSOR TECH'Y	16	369	D	N	400	520	1	4096	S100	Y	S	N	0	4K	5w	16KRA
RE	90D		16	399	D	N	400	520	1	4096	S100	Y	A	N	0	4K	5w	16KRA
MO	90D	S.D. SALES	16	259	D	Y	375	500	1	4115	S100	Y	K	N	0	8K	10.9w	EXPANDO
MO	90D		24	367	D	Y	375	500	1	4115	S100	Y	K	N	0	8K	10.9w	EXPANDO
MO	90D		32	475	D	Y	375	500	1	4115	S100	Y	K	N	0	8K	10.9w	EXPANDO
RE	90D	SEALS	32	750	D	Y	400	400	1	4115	S100	N	A	N	0	4K	3.2w	32KDS
RE	90D		64	1850	D	N	400	400	1	4116	S100	N	A	N	0	4K	3.2w	64KDS
RE	90D	SMOKE SIGNAL BDCSTNG	16	529	S	N	450	-	1	4044	SS-50	Y	A	N	0	4K	10w	M-16A
RE	90D	SPACE BYTE	16	599	S	N	450	-	1	4044	S100	Y	A	N	0	4K	12.8w	16K
RE	90D		16	599	S	N	250	-	1	4044	S100	Y		N	0	4K	12.8w	16K-25
RE	90D	TECH'L DESIGN LABS	16	699	S	D	250	400	1	4200	S100	Y	K	N	0	4K	5w	Z16
RE	90D		16	574	S	D	280	450	2	4050	S100	N	A	N	0	4K	5w	Z16
RE	90D		16	699	D	D	280	400	2	4050-2	S100	N	A	N	0	4K	2.7w	D16
RE	90D		32	869	D	N	250	400	2	4050-2	S100	N	A	N	0	4K	3w	D32

Legend

- MO — Mail Order only
- RE — Sells through Retailers
- D — (Warranty Period) prefixed by number of days
- Y — (Warranty Period) one year, usually assembled units
- D — (Upgradeable) smaller memory capacity available and incrementally upgradeable
- Y — (Upgradeable) upgradeable to larger memory capacity (using different chips, jumpers, etc.)
- * — Memory Access Time and Memory Cycle Time are in nanoseconds
- ** — 1=within one month; 2=between one and two months; 3=beyond two months
- *** — Only the numeric code is listed in indicating the chip used. There should be little difficulty in determining the manufacturer from this code

namic boards and they're relatively less expensive on the whole, but they involve simpler circuitry. Dynamic RAM's require refreshing. Sometimes this refresh is "invisible" to the CPU. That is, the CPU has no idea that refreshing is going on – it takes no part in the refresh. Other boards affect the CPU during refresh by halting all operations which may affect the actual throughput of the microprocessor.

Dynamic RAM chips are, in fact, less expensive than static RAM chips on a chip by chip basis. Yet, the printed circuit board technology is much more expensive. Several dynamic RAM memory boards are designed to be more than the standard two surfaces (one or more planes of circuitry sandwiched in between the top and bottom of the board). This protects the circuitry from excessive noise and other random problems that could aggravate a memory board.

Another important consideration in buying more RAM is whether a memory board can be upgraded to more memory (in small increments or as a single final increment). Give some serious long-range thought to buying one board and adding sockets, chips, and jumpers to upgrade it.

Memory access time also significantly relates to the CPU. If you have a 6502, 6800 or 8080, your system runs at 2 mHz or around 500 nanoseconds. If you have an 8085, it's 3 mHz or about 330 nanoseconds. Z80 CPU's run at 4 mHz or 250 nanoseconds. Keep these speeds and times in mind when checking the price structure in the table.

Note the difference between retail and mail order availability. Mail order may be less expensive, but access to the company may be difficult or immediately impractical. Boards sold through retailers imply service backup and in-hand analyses of the board on a relatively immediate level. Again, you must decide what's more important to you.

The primary chip used in the memory board is listed, though it probably won't affect your decision all that much.

Availability is noted in terms of 0-30 day delivery (=1), 31-60 days (=2), and over 60 days (=3). Give a manufacturer at least four weeks from when you mail in your order and money until delivery of the memory. Most sources listed accept Visa or Master Charge. Nearly all sources, however, encounter some delay when personal checks must clear the bank. If you want memory in a hurry, send a bank money order or certified cashier's check.

With the large number of sources and variations of memory around, you should use caution when assessing your needs and your expectations of a system. Think ahead and you'll do fine.

Most of the boards listed in the table are of the "S-100" bus. Some are of the SS-50 bus (SWTP 6800), some Digital Group, others something else. A few memories (e.g., Prime Radix) are available in S-100 bus format and in an additional format.

If you don't want to lose your program or data, check for memory protect. This feature, also known as "write protect", prohibits data from going into a given memory block once data has been initially entered into it. It's like a read only memory while the memory protect switch is on.

Kits generally cost less than their corresponding assembled version. Semi-kits (from Processor Technology) are partly assembled kits and, logically, cost between the kit price and fully assembled price.

A few companies offer parity checking compatability as an option. Generally software controlled, it simply checks whether the bits read in per byte are of an odd or an even parity. If the parity check conflicts with the byte parity (e.g., the parity bit is odd and the sum of all bits in the designated byte is even), then it detects an error. You must decide the potential and importance of parity in your own system before buying.

Nearly all the boards listed in the chart do not provide for wait states – but the CPU won't require them if you match the appropriate access time to your system's CPU chip. If you have special requirements or slower memory chips, providing for wait states is significant.

Addressing your memory board may be just one contiguous memory using only one initial address. If you have ROM's or PROM's spaced intermittantly throughout the memory, you should skip over these ROM addresses (otherwise you'll waste memory space).

Some boards can be addressed in 4K blocks, allowing for a degree of flexibility in locating memory. Crea-Comp Systems offers their four boards with ROM lockout, a feature that allows you to address a board into memory containing ROM. When the memory senses the ROM, it will skip over it and continue on the other side. This is a great boon for those of us with ROM's floating everywhere in the memory addresses.

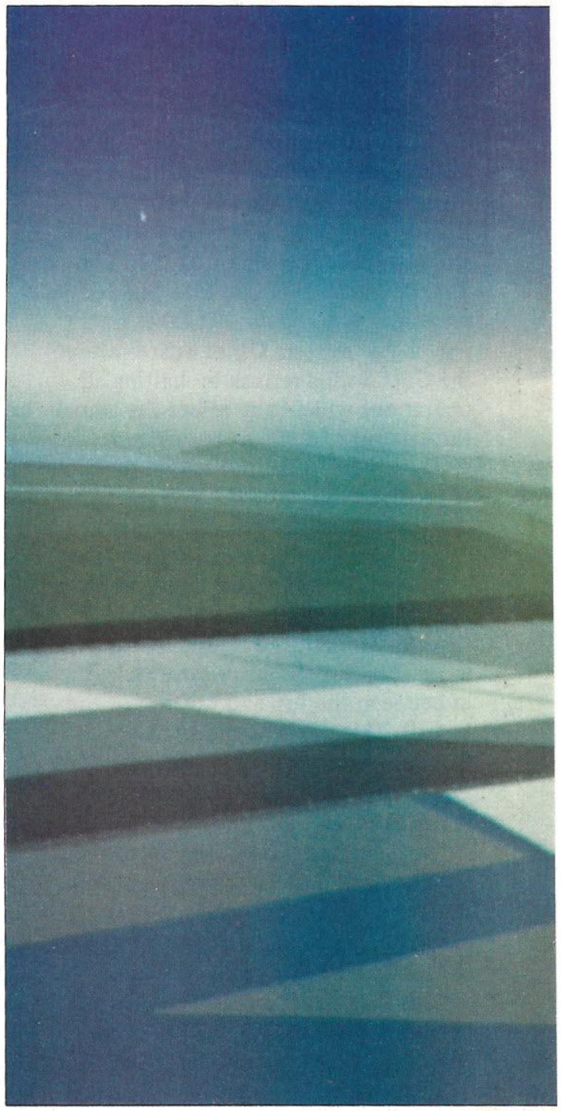
Finally, the power consumption column will give you an idea of how power requirements vary drastically from one board to another. Consider your system's limitations when contemplating a board with high wattage.

Some columns lack data, and other data may change by the time you read this, but it's all close. Once we compile some other comparative charts for your reference you can start your own *Personal Computing* reference guide which you can add to and adjust as the industry and products' specifications change.

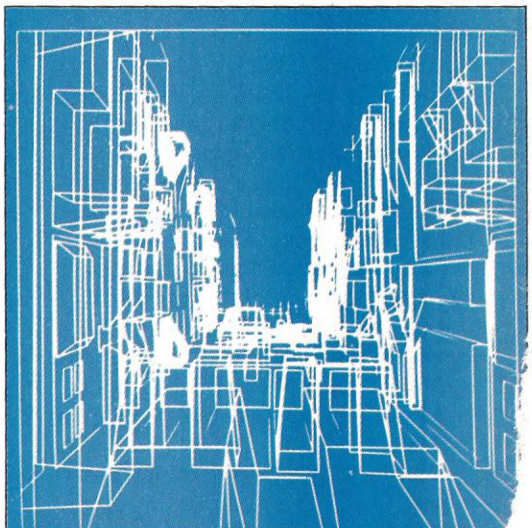
With the large number of sources and variations of memory around, you should use caution when assessing your needs and your expectations of a system. Think ahead and you'll do fine.

Technology and the times: COMPUTER GRAPHICS

BY NEESA SWEET AND VICKY O'HARA



Raster scan graphics, still to costly for home computer sys

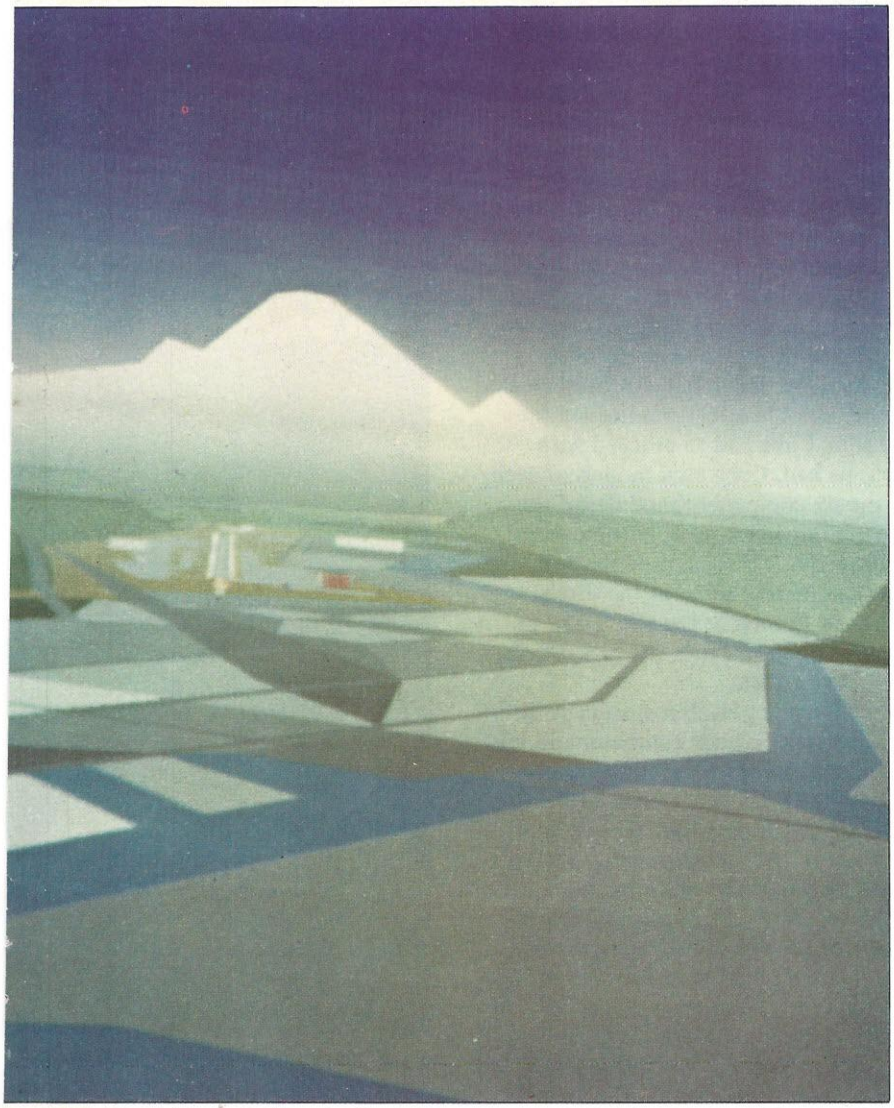


Above: The first realization of the image presents its as a white pattern on a dark background. Right: Ea frame of the Death Star trench was constructed on t cathode ray screen.

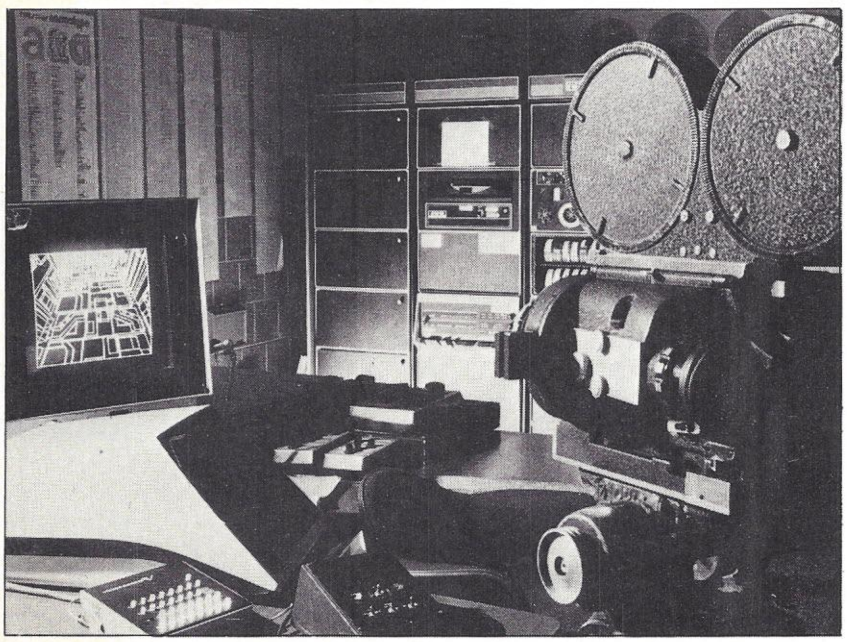
As computer people, we're just coming of age. And computer graphics, which make TV-video games possible, is a fascinating utilization of the computer, our most fascinating toy.

Larry Cuba, a 27-year-old computer filmmaker from Los Angeles, foresees a bright future for computer graphics. Currently a research associate at the University of Illinois, Cuba knows something about computers – he's the guy behind the computer animation sequence in the movie *Star Wars.* He's also worked with the studio which gave birth to the ABC logo, 7-Up commercials and other sequences using computer animation.

Computer graphics have found a place in our everyday lives. The complex systems once used primarily for industrial design have become children's toys on a Saturday afternoon. We are, Cuba said, becoming a "computer people", just as we became a "car people" and a "camera people". Owners of the first cars, he reminisced, had to be mechanics just to survive two-block jaunts to the store. And the first photographers had to make their own plates, mix their own solutions, and develop and print their own film. Then, as technology improved,



tems, produce high resolution color images.



everyone operated cars and the camera became as commonplace as the ballpoint pen.

Computers have followed the same trail. Today, everyone walks around with little calculators in their pockets. Soon, a computer in every home may replace a chicken in every pot. And computer graphics will be part of the change.

Computer graphics serve architects, engineers and particularly the aerospace industry. Companies, such as Boeing, use computer graphics to design complex electrical systems for large aircraft. One of the earlier animated films utilizing computer graphics was used at a Seattle airport to simulate the pilot's view as he landed. With this film as prototype, various cockpit designs were tested for pilot visibility.

Computer graphics have also grown in film and video, rapidly becoming an important, recognized, new art form. In art, Cuba worked extensively with John Whitney, a computer art pioneer, who has been in the film industry since the 40s. Whitney spent years tinkering with mechanical devices, analog control machines, to animate images. Whitney's work led him to digital computers in the 60s.

Whitney's film, "Permutations", was one of the first animated films created by an artist for a non-technical purpose. Cuba calls Whitney the "father of computer art".

Cuba's computer system, which controls the camera, is ideally suited to making films.

In animation, such as cartoon animation, images are put on film and run at a speed of 24 frames per second, creating the illusion of motion. This projection of a series of still pictures at high speed, is the basis for animation. When the camera is integrated into the computer, the tedious task of the animator becomes much simpler.

Computer animation works on much the same principle, with the images programmed first into a computer, then to film. Basically, the system works like this: once the desired image is programmed into the computer, a beam of electrons connects the points by a series of straight lines which appears on the screen of a cathode ray tube (CRT). The first realization of the image presents itself as a white pattern on a dark background. The artist then modifies the design to suit his needs.

continued on following page

When completed, the design may be recorded on film using an interconnected camera, or the computer can mechanically plot the picture on paper, using black or colored inks. The image can be reproduced by rerunning the program and replotting the results. The technique is similar to lithography, where the image is drawn into stone and then reprinted on paper.

Using this method, Cuba created the 40 second animation of the Death Star trench in *Star Wars*. It took months of programming, over 12 hours of shooting time, a computer at the University of Illinois and a computer language called GRASS.

Cuba first became involved with *Star Wars* a year ago January when the film's director, George Lucas, on the lookout for a computer animationist, viewed Cuba's film "First Fig". Cuba submitted a proposal and got the job.

Cuba's scene begins with General Dodonna explaining a dangerous mission to Luke Skywalker and other star-pilots and navigators of the rebel forces. Their mission? Destroy the Death Star. We see Luke and the others assembled in a long, narrow room, watching a computer screening of the one flaw in the Death Star. The General gestures towards the diagram as he explains how the Empire can be attacked.

"You're required to maneuver straight down this trench and skim the surface to this point," the General explains. "The target area is only two meters wide. It's a small thermal exhaust port. The shaft leads directly to the reactor system. A precise hit will start a chain reaction that should destroy the station . . . Man your ships, and may the Force be with you."

The computer schematic helping General Dodonna explain the destruction of the Death Star was actually a piece of film being projected onto the rebels' computer screen. On the screen for only 40 seconds, it was, nonetheless, the result of months of work. It utilized the best in computer hardware technology. It also employed a unique software system that enabled the computer to be used not only for the creation of art but also as a new educational tool.

For Cuba, filming the sequence was a tough problem. His assignment was to simulate the pilot's mission viewed from the approach to the Death Star and through its flight down the trench at its surface. The schematic was to match the reality of the trench itself in all its complexity.

Basically, there are two ways such an effect can be created in film. The classic "blue screen" method, where the rebels actual computer screen would be solid blue in the live action filming to allow later superimposing of the animation sequence, or a "rear projection" method where the film on the screen is preshot and the final sequence is shot as a whole unit. Lucas preferred the second approach to keep a feeling of reality and continuity.

Since the trench had not been created at the time the effect was needed for the film, Cuba had to proceed with what he had. Cuba used models of each of the six component modules of the Death Star trench, and, in some cases, photographs. He also used the computer animation setup at the Circle Graphics Habitat at the University of Illinois, Chicago Circle Campus, Chemistry Department.

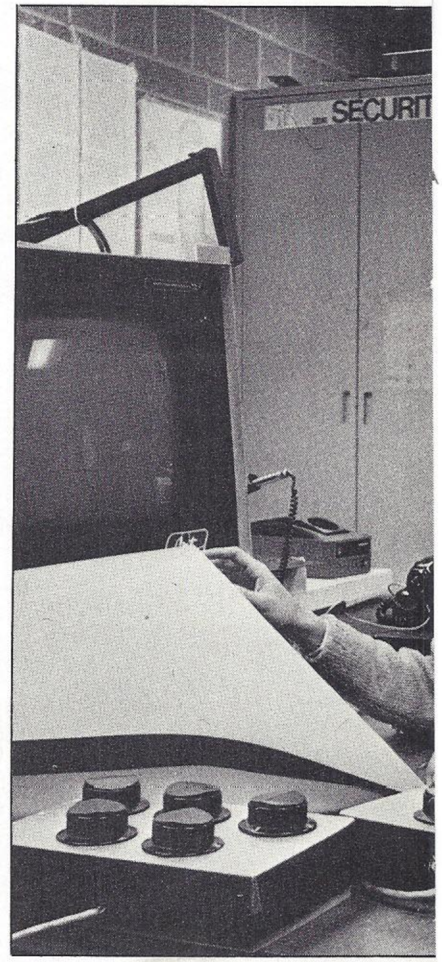
The Graphics Habitat system at the University blends computer and television technology to open the doors of computer creation to educators, artists and others – even if they don't know computer programming. Rather than programming each command, a series of analog instruments allows the operator to form a direct relation between his actions and the image on the cathode ray screen.

The system provides a simple way of creating images that can vary in length, width and height. The system can, over a time span, also change the intensity or color of an image. "It's a fun system to use, like a three-dimensional chalkboard," said Cuba. "And that encourages its success."

At the heart of the system is a Vector General 3DI CRT with dials, light pen and an electronic tablet. Commands fed into the attached minicomputer instruct this device to draw lines between various points. A PDP-11/45 minicomputer performs the actual computations necessary to make pictures.

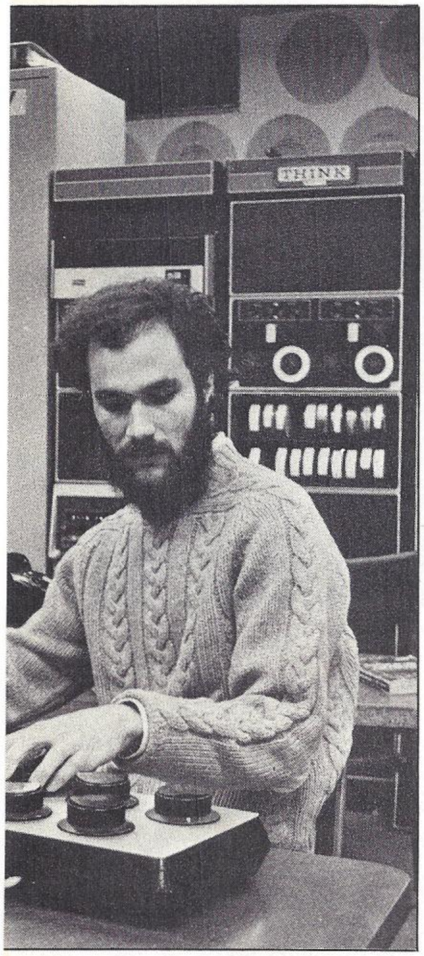
The CRT allows the operator to control the image on the screen. Figures can be called up and manipulated, or images can be traced on the tablet. Electronic impulses sense where the pen is at various points in time. The coordinates of these points are picked up and transferred as an image, with connecting lines, to the screen via the minicomputer.

The Vector General and the mini-



Cuba completed his work at the University Graphics Habitat.

**The 40 second a
Death Star trench
months of programm
shooting time, a
University of Illino
language call**



of Illinois, Chicago Circle Campus,Circle

ımimation of the
in Star Wars took
ıing, over 12 hours of
computer at the
ıis and a computer
ed GRASS.

computer are also hooked up to an "image processor" and other video equipment, all of which allow the operator to place any desired image on tape. In the case of the *Star Wars* sequence, the output had to be on film. This was done directly off the Vector General's screen. The method employed a standard Mitchell 35 animation motor, linked to the comter. The screen and camera nested under a shroud for extra light protection.

The use of GRASS made the filming of *Star Wars* feasible. GRASS (Graphics Symbiosis System) is a computer language developed in 1973 by Thomas A. De Fanti, then a graduate student at Ohio State University. De Fanti's original purpose was to provide a method for artists to explore computer art without the help of a programmer. GRASS provides a system of easy-to-learn commands that perform a variety of tasks with a minimum of repetition.

To demonstrate the language, Cuba produced a graphic representation of the chemical structure of a starch molecule. When the hexagonal molecule appeared on the screen, Cuba rotated it on various axes and even added another stored figure, a propeller. He simulated an airplane flying a path over the screen. Instead of computing the mathematical location of where the starch molecule was and where it had moved, he simply turned dials unitl the image looked right. It is this instant-feedback type of setup – the capability of changing an image with dials and other physical devices – that separates GRASS from other digital concepts and systems.

For *Star Wars*, Cuba ran his program in GRASS with a few additions. On a previous visit to the University of Illinois he had written the tablet program that made possible the use of two-dimensional free-hand drawings on the tablet. The shape of the free-hand drawing became a list of points on the tablet which were then transferred to the computer. The points were stored so that the operator could recall the image from memory at any time. This program was still two-dimensional and not sufficient for entering the coordinates of the Death Star trench models, which had a third dimension, height. However, to rotate the objects on the screen, three dimensions had to be introduced. The tablet program had to be changed. GRASS was capable of working in three dimensions, provided that three dimensions were inserted into the program.

Cuba rewrote the program to allow for this visual magic. Instead of picking up two-dimensional coordinates at every point, the program would stop and wait for a third "Z" (height) to be entered. In this manner, a three dimensional drawing could be placed in the computer and manipulated by GRASS.

Cuba used a photograph of the Death Star (taken from a mat painting) to form the computer picture. The geometric design of the computer picture, consisting of circles and arcs, permitted control by the operator. The picture on the computer screen could be rotated or made larger or smaller by manipulating the dials.

"To complete the shot," said Cuba, "I had to continue the pilot's view of the Death Star to include the flight down the trench to the target at the end. The trench shown in the film was made up of hundreds of models, while six basic shapes formed the 40-foot long trench. I worked from each of the six basic modules and photographs of the trench as it was constructed.

First, a computer representation of each model was made by using numbers to indicate line points. A photograph of each model was first placed on the data tablet. By pressing on the pen, the object being created appeared on the computer screen and a cursor indicated the current position of the pen. The height of the point ("Z" coordinate) was entered separately by selecting one of the function buttons. Another button (reflected in the display) specified whether or not a line was to be drawn between points. To monitor the construction from two to three dimensions, the Vector General dials were used to rotate the picture and examine it from all sides. This process is called "digitizing", which means numbers are used to indicate endpoints of lines.

With the six basic modules stored in his computer, Cuba wrote a program to combine the modules into larger structures. His computer version of the trench was constructed of U-shaped sections placed side by side. "To make one of these sections," he explained. "I recalled a digitized module from memory and positioned it using the dials. Then I added a second module and continued in this way until the 'U' was formed. The completed

Continued on following page

The *Star Wars* pilots assemble in a room to watch a computer screening of the one flaw in the Death Star.

Computer graphics have grown in film and video, rapidly becoming an important, recognized new art form.

figures could then be moved as a unit. To complete the trench, more than 50 U-shaped sections were constructed and stored."

Once the components were completed, each image that was to be part of the finished film had to be shot. Each frame required about two minutes for construction. Each shape was first shown in perspective by the computer and then combined with four others to form a single frame of the film. After completing the computer construction for each frame, the program cued the camera and the exposure was made. Two thousand exposures later, Cuba had the finished trench sequences on film. Two minutes of this time-lapse photography came from about 12 hours of "real time". From the two minutes of time-lapse, only 40 seconds actually appear on General Dodonna's screen.

With the trench problem out of the way, Cuba then left for Chicago to continue his work at the university with two programmers, T.J. O'Donnell and Tom Chomica; a photographer to shoot the models; and a number of University of Illinois students.

Each model had to be carefully measured. Heights were precisely recorded at the correct points so that Cuba would know which height to add to the other coordinates at the moment of filming. The process involved many time-consuming calculations. It also involved slow presentation of many pictures that had become complex with newly-constructed vectors. Time was running short and the program still needed to be debugged.

As the deadline for completing the sequences approached, the situation became tense. Then, the hardware started breaking down. They tried everything. They removed the disk, cleaned it, took it apart and each time, 30 minutes later, CRASH! Something would not let the program complete its run.

After working feverishly up until two days before deadline, Cuba decided to call Los Angeles on Monday and tell them he was having equipment problems. They would have to go with the blue screen method – there was no longer any choice. Just as he was about to give up, Cuba accidently found the solution. The air conditioner was foul-

ing up the hardware. One of the computer circuits was too sensitive. It could not stand the consistently low temperatures under which computers work. Turning off the air conditioner during the computer run allowed the computer to finish without breakdown. The film ran without a hitch all the next day and was shipped out in time for the final shooting.

In the end, the computer-animated sequence was impressive. But Cuba doesn't think such complicated photography will start an avalanche of computer animation in feature films. He does insist, though that there's a valuable lesson to be learned from *Star Wars,* a lesson which should make computer graphics an integral part in the evolution of entertainment.

Most feature films, he explained, are dramas based on basic confilicts and resolutions. "So, if you already know the ending," he says, "you can't really enjoy it beyond that point. But if you go to a museum to see an abstract painting, you can see it many times without knowing how the painting comes out. You go there to enjoy a visual experience, the same way you might listen to a record or a piece of music many times even though you know how the finale sounds.

"And *Star Wars*," he continued, "doesn't seem to be hurt by repetition. They get that thrill from the explosions, the sounds – everything that's going on visually and aurally. They get the same excitement every time they see the film even though they know who's going to win in the end."

Cuba says this lesson, coupled with the refinement of videodiscs – which will allow the public to watch films of their own choosing on television – will create a demand for more films like *Star Wars* that provide visual and aural experiences long after the bad guys are dead. He believes that, in the future, people will buy videodiscs the same way they now buy records. People will demand something enduring for their money. And therein lies a vast potential for computer graphics and for the people involved in the computer sciences.

For the first time, computer graphics will become accessible to the masses as an art form. And Cuba believes that computer people eventually will enshrine the art of our age in museums and libraries, alongside the once bastard, and now legitimate, photography.

PLAY STAR WARS

BY JERRY SELF

Trust the Force. Go with your instincts and destroy the Deathstar. Fly Luke Skywalker's fighter ship against the ultimate weapon of the Grand Moff Tarkin. Try playing STAR WARS!

As in the movie, the task at hand involves bombing a small shaft in the Death Star before the Death Star destroys the moon on which the rebel base is located. The bomb trajectory appears as a trail of descending asterisks. The bomb has a range of six spaces.

Bombing the shaft within the ten turns allotted is actually rather easy. Easy except for contending with Imperial Fighters. If you don't destroy the Death Star or eliminate the fighter within two turns, the Imperial Fighter will shoot you down. Unless, of course, Han Solo in his Millenium Falcon comes to the rescue.

You can adjust the Death Star grid. Use a 10, 20 grid for an easy beginner's game. A more challenging 20, 60 grid just fills a 24-line CRT. Experimenting with short and wide or tall and skinny grids brings some unusual results.

For the more challenging sizes you might want to add:
960 ON Z GOTO 965,970,980
965; "IMPERIAL FIGHTER CLOSING IN" & Z=Z+1 & GOTO 1140. That will give you a third chance to hit the fighter.

Imperial Fighters are randomized to show up about half the time. Thirty percent of the time, in random pattern, Solo will destroy them for you. Changing the ".5" in line 910 will control the percentage of appearances of Imperial Fighters. The ".3" in line 250 affects the rescue rate of Han Solo. The FOR J=1 TO 5 instruction in line 320 controls the range of the

Continued on following page

Star Wars game program

```
10 * STARWARS
20 *    BY JERRY SELF
30 *    OCTOBER 3, 1977
70 ; "WHAT SIZE GRID DO YOU WANT?  NOVICE=20,20 OR CHALLENGE=20,40."
80 ; "LIMITS ARE 60,60.  SIZE"; & INPUT S1,S2
90 W,T1,T2=0
100 ; "DO YOU WANT INSTRUCTIONS Y/N";
110 INPUT A$ & IF A$="Y" THEN 520
120 DIM G(60,60)
125 * POSITIONING THE TARGET
130 C1=INT(RND(1)*S1) & IF C1=0 THEN 130
140 C2=INT(RND(1)*S2) & IF C2=0 THEN 140
150 T2=T2+1 & F,M,Z=0
155 * INITIALING THE GRID
160 FOR I=1 TO S1
170 FOR J=1 TO S2
180 G(I,J)=" "
190 NEXT J & NEXT I
200 G(C1,C2)=2 & N=11
205 * THE PLAY BEGINS
210 FOR I=1 TO 10
220 N=N-1 & ; N;"MINUTES TIL DEATHSTAR CLEARS PLANET."
230 GOSUB 900
235 * RANDUMIZING HAN SOLO TO THE RESCUE
240 IF Z>0 THEN 250& GOTO 280
250 M=INT(RND(0)+.3) & IF M=1 THEN 260& GOTO 280
260 G(C3,C4)=" " & Z=0
270 ; "HAN SOLO HAS ELIMINATED THE IMPERIAL FIGHTER."
275 *   TARGETING THE BOMB
280 ; & ; "ENTER COORDINATES"; & INPUT X & INPUT Y
290 IF X<1 THEN 280& IF X>S1 THEN 280& IF Y<1 THEN 280& IF Y>S2 THEN 280
300 IF G(X,Y)=2 THEN 420 & IF G(X,Y)=4 THEN 710 & G(X,Y)=1
315 * CHECKING FOR STRIKE
320 FOR J=1 TO 5
330 IF X+J>S1 THEN 360& IF Y+J>S2 THEN 360
340 IF G(X+J,Y+J)=2 THEN 460& IF G(X+J,Y+J)=4 THEN 710
350 G(X+J,Y+J)=1
360 NEXT J
370 GOSUB 740
380 NEXT I
385 * REPORTING OUTCOME OF GAME
390 ; "THE DEATHSTAR HAS DESTROYED THE REBEL BASE."
400 ; "YOU ARE HOMELESS AND LOSING POWER." & W=-10
410 ; "DARTH VADER WILL GET YOU." & GOTO 480
420 G(X,Y)=3 & GOSUB 740
430 ; "!!!IMMEDIATE DIRECT HIT!!!  YOU HAVE DESTROYED THE DEATHSTAR"
440 ; "AND SAVED THE REBEL CAUSE!" & W=N*10
450 GOTO 480
460 G(X+J,Y+J)=3 & GOSUB 740
470 ; "YOU HAVE DESTROYED THE DEATHSTAR!" & W=N*5
480 T1=T1+W & ; "SCORE THIS GAME";W,"TOTAL";T1,"AVERAGE";T1/T2
485 ; "DO YOU WANT TO PLAY AGAIN Y/N";
490 INPUT A$ & IF A$="Y" THEN 130
500 ; "MAY THE FORCE BE WITH YOU"
510 END
515 *   INSTRUCTIONS
520 ; "THE DEATHSTAR HAS THE POWER TO DESTROY A PLANET WITH A"
530 ; "SINGLE BLAST.  IT IS APPROACHING THE REBEL BASE WHICH"
540 ; "IS ON A MOON HIDDEN BEHIND THE MOTHER PLANET.  AS SOON AS"
550 ; "THE DEATHSTAR CIRCLES THE PLANET IT WILL DESTROY THE BASE."
560 ; "YOU ARE FLYING A FIGHTER WHICH CAN BOMB A SMALL"
570 ; "SHAFT LEADING TO THE POWER SOURCE OF THE DEATHSTAR."
580 ; "IF YOU HIT IT YOU WILL DESTROY THE DEATHSTAR AND SAVE"
590 ; "THE REBEL BASE."  & ;
600 ; "ENTER COORDINATE X (WITHIN GRID LIMIT)  AND Y TO TARGET"
610 ; "YOUR BOMB.  THE BOMB TRAJECTORY WILL THEN BE SHOWN"
620 ; "ON YOUR SCREEN.  THE TARGET (#) WILL ALSO APPEAR."
630 ; "YOU CAN THEN ESTIMATE HOW MUCH YOU MISSED.  THE FIGHTER"
640 ; "IS EQUIPPED WITH A COMPUTERIZED GRID WHICH CAN SHOW THE"
650 ; "EXACT LOCATION OF THE TARGET.  BUT YOU DO NOT USE IT"
660 ; "BECAUSE YOU TRUST THE FORCE."
670 ; "BUT WATCH OUT FOR IMPERIAL FIGHTERS!"
680 ; "IF YOU OR HAN SOLO DO NOT ELIMINATE THEM WITHIN TWO TURNS,"
690 ; "THEY WILL ELIMINATE YOU.  MAY THE FORCE BE WITH YOU!" & ;
700 GOTO 120
705 *   REPORTING STRIKE ON FIGHTER
710 G(C3,C4)=" " & Z=0
720 ; "YOU HAVE HIT THE IMPERIAL FIGHTER." & GOTO 370
735 *   PRINTING THE GRID
740 FOR K=1 TO S1
750 FOR J=1 TO S2
760 IF G(K,J)=1 THEN 810 & IF G(K,J)=2 THEN 820
770 IF G(K,J)=3 THEN 860 & IF G(K,J)=4 THEN 830
800 GOTO 840
810 ; TAB(J)"*"TAB(0) & G(K,J)=" " & GOTO 840
820 ; TAB(J)"#"TAB(0) & GOTO 840
830 ; TAB(J)"X"TAB(0) & GOTO 840
840 NEXT J & ; & NEXT K & GOTO 890
860 ; "*************************************************************"
870 ; "***********************KABOOM********************************"
880 ; "*************************************************************"
890 RETURN
900 IF Z>0 THEN 960& *ROUTINE TO RANDOM IMPER. FIGHTER
910 F=INT(RND(0)+.5) & IF F=1 THEN 920& GOTO 1010
920 C3=INT(RND(1)*S1) & IF C3=0 THEN 920
930 C4=INT(RND(1)*S2) & IF C4=0 THEN 930& Z=1
940 G(C3,C4)=4
950 ; "IMPERIAL FIGHTER ON THE ATTACK" & GOTO 1140
960 ON Z GOTO 970,980
970 ; "IMPERIAL FIGHTER PENETRATING SHIELDS" & Z=Z+1 & GOTO 1140
980 ; "THE IMPERIAL FIGHTER HAS DESTROYED YOUR SHIP."
990 ; "THE DEATHSTAR WILL DESTROY THE MOON AND THE REBEL BASE."
1000 GOTO 480
1005 *  PRINTING RADIO MESSAGES
1010 ON N GOTO 1020,1040,1050,1060,1070,1090,1100,1110,1120,1130
1020 ; "THE GRAND MOFF TARKIN HAS ORDERED THE DESTRUCT SEQUENCE"
1030 GOTO 1140
1040 ; "DARTH VADER IS DETERMINED TO GET YOU" & GOTO 1140
1050 ; "THE REBEL CAUSE DEPENDS ON YOU" & GOTO 1140
1060 ; "TRUST THE FORCE" & GOTO 1140
1070 ; "THE TRADITION OF THE JEDI KNIGHTS DEMANDS THAT YOU WIN"
1080 GOTO 1140
1090 ; "REMEMBER THE JAWAS" & GOTO 1140
1100 ; "SOLO'S MILLENIUM FALCON IS STANDING BY" & GOTO 1140
1110 ; "R2D2 IS CORRECTING A MALFUNCTION" & GOTO 1140
1120 ; "C3PO HAS BROKEN INTO A SWEAT" & GOTO 1140
1130 ; "PRINCESS LEIA RADIOES 'PLEASE HURRY'" & GOTO 1140
1140 RETURN
```

Note – 1000 should read 1000 W=N (-10) & GO TO 480.

bomb. Change the "5" to "2" or "3" for a more exciting game.

Lines 810, 820 and 830 fascinate me. These lines control the printing of the bomb trajectory, the placement of the target, and the position of the X-shaped Imperial Fighter. I'll let you analyze for yourself what the value of this instruction might be.

810 ; TAB(J) "*"TAB(O). Consider what would happen if the target, a fighter, and the path of your bomb all were on the same line and the instruction were simply: 810; TAB(J)"*".

Try your hand against Darth Vader and may the force be with you. ▮◀

A typical game of STAR WARS might read like this:

```
WHAT SIZE GRID DO YOU WANT?
NOVICE=20,20
OR CHALLENGE=20,40.
LIMITS ARE 60,60. SIZE ?20, 30
DO YOU WANT INSTRUCTIONS Y/N ?N
PRINCESS LEIA RADIOS 'PLEASE HURRY'
ENTER COORDINATES ?5, 5

*
 *
  *
   *
    *
     *

9 MIN. TIL DEATHSTAR CLEARS PLANET
IMPERIAL FIGHTER ON THE ATTACK
ENTER COORDINATES ?1, 21
         *
          *
           *
            *
             *
              *

8 MIN. TIL DEATHSTAR CLEARS PLANET
IMPERIAL FIGHTER PENETRATING SHIELDS
ENTER COORDINATES ?3, 21

************************************
**************KABOOM************
************************************

!!!IMMEDIATE DIRECT HIT!!! YOU HAVE
DESTROYED THE DEATHSTAR AND
SAVED THE REBEL CAUSE!
SCORE THIS GAME 80    TOTAL 80
AVERAGE 80
DO YOU WANT TO PLAY AGAIN Y/N ?N
MAY THE FORCE BE WITH YOU
   510 HALT
```

Note on TAB(O): This doesn't work on all computers.

Beginner's Guide to Computer Jargon

BY JOHN T. SHEN

Human nature being what it is, we're always trying to develop tools to make problem-solving easier. We also try to develop tools to do monotonous and mechanical jobs so we'll have more free time to do the things we enjoy. So, unlike humans, the tools we develop make fewer mistakes, work without getting tired and don't go on strike.

One of the best tools we've created is that creature called "computer". But what is a computer? What's the difference between a computer and a microcomputer? What do we mean by multiprocessing? And what do we mean by large scale integration (LSI)?

A computer is an electronic tool that can accept information supplied by a human or another machine. A computer also accepts instructions regarding what to do with the information supplied. The computer then performs the operations on the given information. After the instructions are performed, the computer supplies the results to the person who requested them, or to another machine which may need the results to carry out other operations.

A basic computer is usually composed of an input and output (I/O) unit, memory (or "storage") unit, arithmetic and logic unit (ALU) and a control unit.

The input unit accepts the information to be operated on from people or other machines, and the output unit makes the results available in terms a human can understand.

The memory unit stores information until needed by one of the other units, such as the artithmetic and logic unit, the control unit or the I/O unit.

The arithmetic and logic unit (ALU) does the arithmetic and logic operations necessary to sort or search for particular items or perform mathematical procedures.

The control unit manages all the other units. For example, the control unit decides when the I/O unit will accept information and when the information should be sent from the I/O unit or the memory unit to the ALU for processing. The control unit also decides what operations to carry out and in what sequence. When an operation is completed, the control unit decides whether the results should be sent to the I/O unit or the memory unit.

The technology for building today's computers is very different from the technology that built computers 10 to 20 years ago. Unitl the late 1950's, computers were built from electronic tubes, mechanical relays, resistors and capacitors. We call these computers the "first generation".

From the late 1950s to the early 1960s, computers were built from discrete transistors, resistors and capacitors. We call these computers the "second generation".

In the early 1960s, a new technology arose, called integrated circuits (IC), where many components are fabricated on a chemical substrate called a "chip", which is about 1 centimeter square. In the early 1960s only 100 transistors could be packed on a chip. Computers implemented with 100-transistor chips are called the "third-generation".

Later, new fabrication techniques were developed, so that today we can pack 1000 or more transistors on a single chip. We call these computers "fourth generation".

From the first to the fourth generation, the physical size of a computer with the same computing power shrank drastically. The cost also decreased impressively, and the computing speed increased several magnitudes.

A computer designed for use in many fields of business and science is called a "general-purpose computer". A computer designed for a specific purpose, such as monitoring a patient's heart condition, is called a "special-purpose computer".

A small computer is called a "minicomputer". A *very* small computer is a "microcomputer". The central processing unit (CPU) of a microcomputer is called a "microprocessor".

If a computer has more than one CPU, and if the CPU's are operating in parallel, the computer is called a "*multi*processing computer" or a "multiprocessor".

But just having a computer will not solve your problems. You need a way to instruct the computer to solve a problem. One way is to write a "program" (a set of instructions or steps that tell the computer exactly how to solve a problem).

Since English is our native language, the languages used for writing programs are usually English-based, examples of English-based languages are FORTRAN, COBOL and ALGOL. Some of the programming languages are mathematically-oriented, such as APL. Both types are "human understandable". They are called "high-order languages".

But all computers are built on the simple language of "yes" and "no" (or 1's and 0's), which is the "machine language". To use high-order languages, we must build translators to act as interpreters between man and machine.

The programs programmers write to solve their particular problems are called "application programs". The large program developed by the computer manufacturer for managing the computer resources such as I/O devices, memory spaces and CPU time is called the "operating system". Programs that facilitate the easy use of I/O devices and peripheral memory are called "utility programs".

The application programs, language translators, operating system and the utility programs are called "software".

When a language translator completely translates a program before the execution of that program the translator is called a "compiler". If a translator translates one statement of a program at a time and executes that statement immediately, the translator is called an "interpreter".

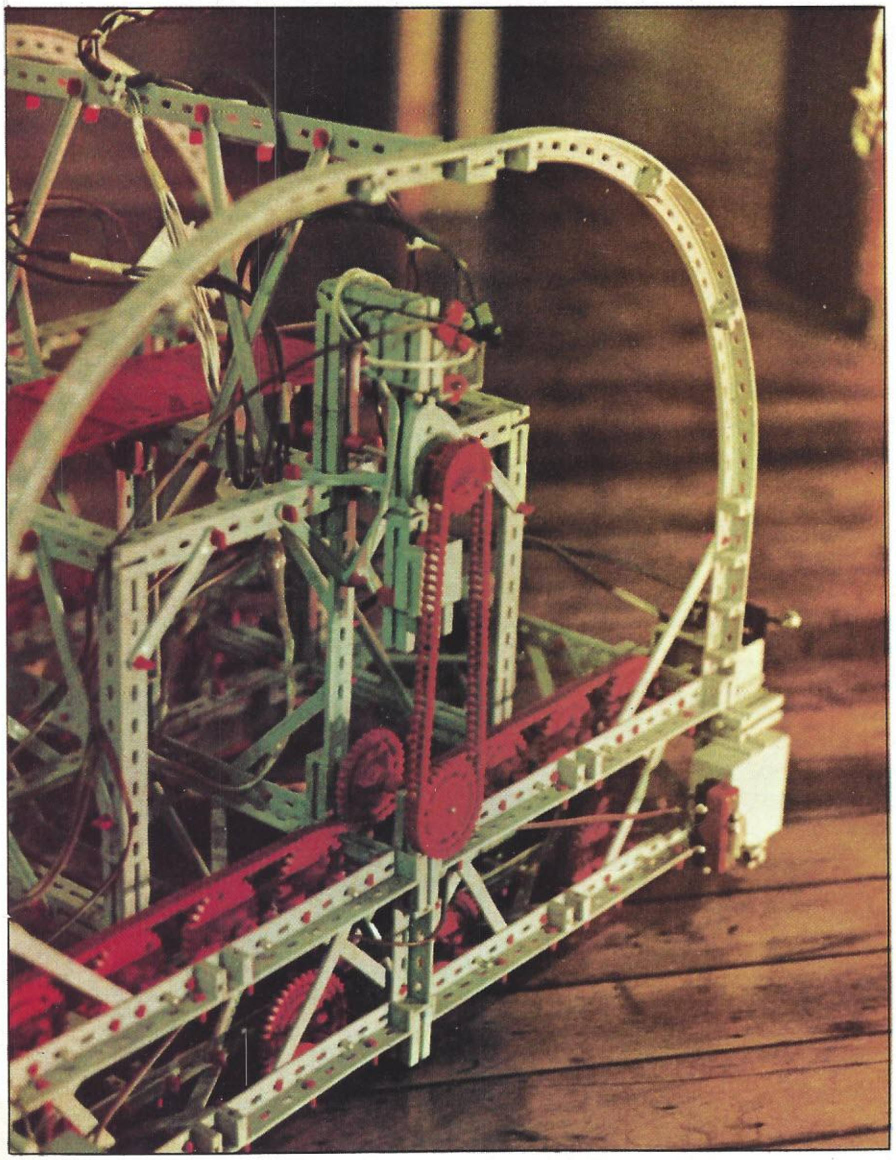
When a portion of a control unit is electrically programmed into a device call "read-only memory" (ROM), or when some of the software is electrically programmed into one or several ROMs, the programming is called "microprogramming".

Building Your BASIC Robot

BY SAM NEWHOUSE

Robots vary in sophistication and control structure, from Mars-walking space probes and industrial assembly robots to self-motivating household pets and computer-controlled "turtle" drawing robots.

My interest in these simple mobile computer input-output devices, robots, led me to construct my own "mechanical pet", Bert. Bert can move forward or backward, and pivot in either direction. Through a separate computer-controlled monitoring system, he responds to a pre-programmed direction after bumping into something and records his steps on film through a mounted video camera. A distance sensor controlled by the same computer keeps track of the distance he travels and guides him on a pre-programmed path.



Bert stops on command for any set length of time, and on command resumes his program where he left off. My reason for building this "robot" was, from the beginning, to construct a peripheral for my SWPC 6800 computer. My goal has been achieved – Bert is controlled entirely by SWPC 8K BASIC, version 2.0 And although there are speed limitations with BASIC, careful programming and reasonable compromise allow for the convenience of BASIC real-time control.

Bert is constructed almost totally of replaceable, interchangeable plastic parts manufactured by the fischertechnik Company of West Germany. Although expensive, the building material allows for quick translation of ideas into physical reality, and easy modification of an existing design.

Struts of various sizes, nylon/plastic equivalents of I-beams, rivets which attach struts to I-beams or other struts, plastic chain, gears, metal axles, cotter pins, motors with reduction gears and worm gears make up the building materials. Flexible skin-like material can also be attached to I-beams with rivets.

These materials result in structures that are light but strong. Bert can move 50 lbs. with only minor reduction in speed – Bert only weighs about 10 lbs.

Eight parallel fischertechnik low-voltage motors drive the robot (4 motors drive each of the two tracks). The motors are driven by regulated adjustable power supplies. The power now in use travels to Bert via cables at about 1.5 amps per side at 12 volts. A third power supply drives the CRC express serial relay interface and delivers .5 amps at 5 volts.

The CRC interface connects to the control computer via a serial output

port. By sending ASCII characters to the interface, followed by an execute character, the relays change position, and power is supplied to specific motors in the robot.

For example, by sending the ASCII sequence "A!E!C!G!" to the interface, both tracks will move forward. The A!E! commands the two direction relays to supply forward voltage. The C!G! sets both sides to "on". A forward motion is accomplished simply by the program statement: print "A! E!C!G!".

At present, the robot may be programmed under BASIC to move forward or backward, or to rotate left or right. During rotation, one track moves forward, while the other is in reverse. During a continuous rotation the robot does not move from its initial position more than a few inches.

Inputs to the control computer consist of 5 bits of a parallel input port. These consist of 4 bits, 1 each, from the front, back, left, and right sensor arrays, and 1 bit for distance travelled detection.

A gear rides on the chain which forms the tread of the robot's track. This gear drives another chain which in turn drives a rotating elliptical gear. At the top and bottom of its rotation, the elliptical gear strikes two micro switches which are wired in parallel. Therefore, each rotation of the gear sends two pulses to the computer port.

The 6800 MPU uses what is called memory mapped input/output. This means that an input port, to the computer, appears as a memory location, just like any other location. The five inputs are, in effect, wired right into the memory so that whenever you want to "scan" this port, all you do is take a "PEEK" at its location, using BASIC.

The PEEK is a BASIC language function, such that PEEK (J) will return the decimal value of the contents of memory location J. To read the input port you just use the PEEK statement. No handshakes or interrupts are necessary.

A tight BASIC wait loop counts the distance pulses by constantly "PEEKING" the location where the input data appears. Since BASIC is slow, and these pulses are of short duration, a certain percentage of distance pulses are missed.

The same routine that counts distance pulses also constantly checks for a sensor closure – an interruption in the robot's control program that occurs when these "sensors", which are microswitches mounted on the front, back, and sides of the robot, are closed.

This self taught tinkerer built his own robot and programmed it using BASIC.

The robot then moves in a direction opposite to the original direction, upon hitting an obstruction.

The inputs are arranged so that the distance pulse (I0) represents the least significant bit. Front sensors are I1, back are I2, left I3 and right I4.

At any time, "PEEKING" the input port gives a number (decimal) between 0 and 255. If no sensors show a "hit", and if no distance pulse is present, the port will show 255_{10}. However, if any sensor shows a hit, then the input port will show a number less than 254, regardless of the status of the distance pulse.

The input routine first looks for a sensor hit, then a distance pulse, and finally, for a second sensor hit. Any time a sensor hit registers, the computer goes into its interrupt subroutine, which stops the motors, and then reverses them for a fixed period of time. During this brief time the robot is not responsive to further sensor hits.

The BASIC control program recognizes the following commands: FX, BX, RX, LX, SX, and JX. The robot moves forward, backward, left, or right X number of pulses by FX, BX, LX, RX respectively. SX stops the robot for a time period proportional to X. If X = 0, control is immediately returned to the options menu portion of the program. JX loops control to Step Number X. This is usually used to repeat a pattern over and over by "jumping" back to the initial program step repeatedly.

The control program will also allow robot control patterns of up to 50 steps to be saved and loaded from paper tape. A list of the current program may also be obtained.

The control program is written in a top-down, structured programming style, and is easy to modify. If the robot were programmed in assembly or machine language, its control would be much more precise – no distance pulses would be lost because of the slow speed of BASIC.

Tests have shown that the control program and robot are precise enough (as currently developed) to bring the robot to the same spot after executing a short stored program, plus or minus about 1 foot. This means that, if the robot is placed in a particular spot, pointing in a known direction, execution of a stored program will leave it no farther than about 1 foot from where the same stored program drove the robot previously.

Another application of the robot is to carry a standard video camera. In this mode, the robot can be directly controlled by a SWTPC joystick. Again, a program coded in BASIC is used to interface the joystick to the CRC serial

Technical Specs on Robot Package

H x W x L -	18" x 19" x 26"
Weight -	25 lbs. including camera
Speed -	1.5 mph at 12 volts
Motors -	8 – fischertechnik motors with gearing
Power -	2 – 1.5 ampere variable voltage regulated power supplies for motors
Supplies -	1 – 1.5 ampere 5 volt power supply for relays
Camera -	Sony AVC3200 video camera
Computer -	SWTPC6800 – 16K memory
Software -	SWTPC 8K BASIC, Version 2-0
Peripherals -	Slot 0 – serial port to relay interface
	Slot 1 – serial control terminal
	Slot 3 – parallel interface to joystick
	Slot 5 – parallel interface to sensors and distance pulses
	Slot 6 – serial interface for Teletype
Sensors -	3 – forward – wired in parallel
	2 – right "
	2 – left "
	5 – rear "
Relays	CRC Xpres serial control interface – contains 4 SPST relays

interface which, in turn, controls Bert's motors.

The human operator sits in front of a TV monitor which shows what the robot-mounted TV camera is pointing at. By moving the joystick gently forward, the robot will respond by moving forward. Move the stick back to center, and the robot stops. Move the stick right or left, and the robot rotates right or left.

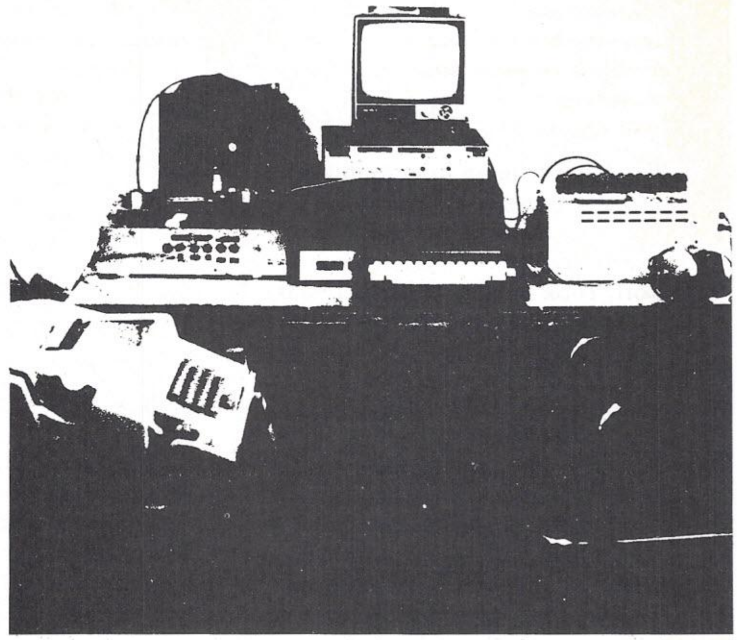
A signal transmitted by the camera goes through to a video recorder before it reaches the monitor. A scene could be set up and videotaped by the moving camera. By going forward or backward, slow zoom-ins are possible.

At this time, the camera cannot tilt up or down. As it approaches an article on the floor, the object disappears from the camera's view.

My plans for the future include constructing a "camera tilter" from fischertechnik parts, proximity sensors, an on-board computer and a radio link to the SWTPC computer.

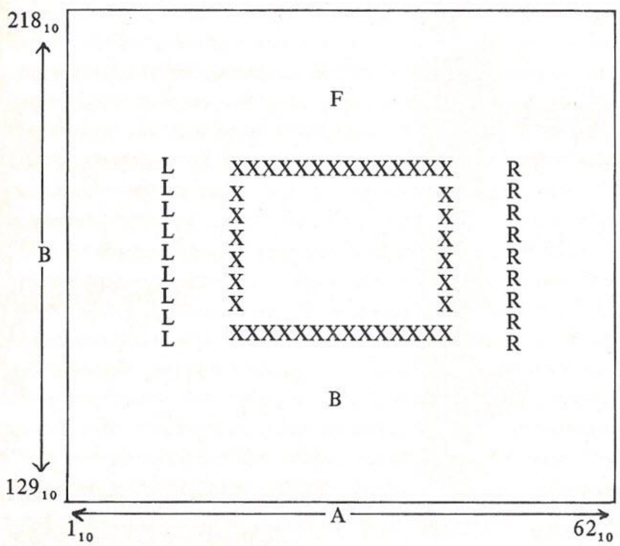
The onboard computer will follow its own built-in program, responding to sensor hits and counting distance pulses. The radio link will enable the robot to carry out instructions sent out by the remote master computer. These instructions would come in a general rather than specific form; examples might include patrol, sleep, wait (until

How the Robot Program Works



When you have constructed your robot, you can get him moving with these BASIC programs. First, though, you must understand the CRC Xpres Relay Interface before you can comprehend Program I. This board contains four relays and

Screen Diagram



Within the square outlined by the X's, the robot will stop. The cursor must be moved close to the "F" or "B" for motion to begin. With "L" and "R", it is sufficient for the cursor to be behind the line of L's or R's, and not above or below the ending L's or R's.

connects to a serial port at Slot Ø. Any relay may be turned off, one at a time, as follows:

	On	Off
Relay 1	A	B
Relay 2	C	C
Relay 3	E	F
Relay 4	G	H

The execute character to the relay board performs the most recent command received. The execute character is the exclamation point (!). Sending "E!" to Port 0 would turn on Relay 3.

Relay 1 controls a DPDT power-relay to the right side. This controls the motors forward or backward.

Relay 2 supplies power to Relay 1.

Relay 3 activates a DPDT reversing relay that controls the left side.

Relay 4 supplies power to Relay 3.

Thus, PRINT #0, "A!E!C!G!" would turn on all relays and the robot would march forward.

The commands implemented so far which control the robot motors are:

Meaning	Forward	#	Comment
Forward	F	X	Where X is the # of the pulses
Backward	B	X	"
Left	L	X	"
Right	R	X	"
Jump	J	X	Where X is the control statement
Stop	S	X	Where X>0 stop for X time periods
or	S	0	Return to options menu
Recorder On	V	X	Where X can be anything
or Off	X	X	"

Another relay activates the video recorder. This relay functions through the SWTPC AC-30 Cassette Interface Board's "Cassette Motor Control Output". Sending the appropriate control character to the CT-1024 terminal, opens or closes the relay. Control T turns on the relay, Control R causes it to turn off.

motion detector is activated) or program.

The program instruction conditions the on-board computer to accept a series of specific motion control commands from the SWTPC computer, store these commands, and execute them until an override command is received via the radio link. Examples of specific motion control commands are: forward 10 pulses, right 5 pulses, stop 3 time periods.

The radio link will be made from a Citizens Band Class "C" tone transmitter and receiver. This transmitter will be modulated, or turned on and off, by the SWTPC 6800. All output will appear in the form of serial ASCII characters. At the receiver, a waveform corresponding to the original ASCII character will be produced, which will be fed through a level converter to the serial input port of the on-board computer.

Miniature lead-acid batteries will provide power to the motors. Two batteries will power each track (see photos). The on-board computer will be powered by 7 amp/hr Gould nicads.

What the purpose of all this is, I'm not sure. I had a lot of fun building and designing my robot and I plan to go a lot further. My fascination with robots will continue unabated. This simple project opened my eyes to the incredible complications that can pop up when you try to design a robot.

Program I

```
0001  F4=32790
0002  F5=254
0003  F1=1
0004  F0=0
0005 DIM A$(50),B(50)
0006  F2=2
0007  F3=3
0008  F9=4
0009  F8=2
0010 PRINT #1,CHR$(16);CHR$(22);
0020 PRINT #1,"       OPTIONS"
0030 PRINT #1,"1 - PROGRAM"
0040 PRINT #1,"2 - RUN"
0050 PRINT #1,"3 - SAVE"
0060 PRINT #1,"4 - LOAD"
0070 PRINT #1,"5 - END"
0075 PRINT #1,"6 - LIST"
0078  P2=6
0080 PRINT #1
0090 PORT= 1
0100 INPUT "#",C
0105 IF C>6 THEN 10
0110 IF C<1 THEN 10
0115  C=INT(C)
0120 ON C GOSUB 1000,2000,3000,4000,5000,6000
0130 GOTO 10
1000 REM *** PROGRAM SECTION
1010 PRINT #1,CHR$(16);CHR$(22);"MAXIMUM
     NUMBER OF STEPS IS 50"
1020 PRINT #1,"MAXIMUM DIRECTION IS 50"
1022 PRINT #1,"ENTER '#' WHEN PROGRAM DONE"
1030  I=1
1035 PRINT #1,"STEP #";I;
1040 INPUT A$(I)
1050 IF A$(I)="#" THEN RETURN
1060 INPUT "DISTANCE",B(I)
1070 IF B(I)>50 THEN 1060
1090  I=I+1
1100 GOTO 1035
2000 REM *** RUN SECTION
2010  I=F1
2020  A$(I)=LEFT$(A$(I),1)
2025 IF A$(I)="F" THEN 2200
2030 IF A$(I)="B" THEN 2300
2040 IF A$(I)="L" THEN 2400
2050 IF A$(I)="R" THEN 2500
2060 IF A$(I)="S" THEN 2600
2070 IF A$(I)="J" THEN 2700
2075 IF A$(I)="V" THEN PRINT"";:I=I+1:GOTO2020
2077 IF A$(I)="X" THEN PRINT"";:I=I+1:GOTO2020
2085 PRINT #1," *** ERROR AT STEP #";I
2090  I=I+1
2100 GOTO 2020
2200 REM *** F ROUTINE
2210  T=F0
2215  D=F1
2217 PRINT #0
2220 PRINT #0,"A!E!C!G!A!E!C!G!A!E!C!G!";
2230  Q=PEEK(F4)
2235 IF Q<F5 THEN GOSUB 7000
2240 IF Q<>F5 THEN 2230
2245 IF Q<F5 THEN GOSUB 7000
2250  T=T+F1
2260 IF T>B(I) THEN 2270
2265 GOTO 2230
2270  I=I+F1
2280 GOTO 2020
2300 REM *** BACKWARD ROUTINE
2310  T=F0
2315  D=F2
2317 PRINT #0
2320 PRINT #0,"B!F!C!G!B!F!C!G!B!F!C!G!";
2330  Q=PEEK(F4)
2335 IF Q<F5 THEN GOSUB 7000
2340 IF Q<>F5 THEN 2330
2345 IF Q<F5 THEN GOSUB 7000
2350  T=T+F1
2360 IF T>B(I) THEN 2370
2365 GOTO 2330
2370  I=I+F1
2380 GOTO 2020
2400 REM *** LEFT ROUTINE
2410  T=F0
2415  D=F3
2417 PRINT #0
2420 PRINT #0,"B!E!C!G!B!E!C!G!B!E!C!G!";
2430  Q=PEEK(F4)
2435 IF Q<F5 THEN GOSUB 7000
2440 IF Q<>F5 THEN 2430
2445 IF Q<F5 THEN GOSUB 7000
2450  T=T+F1
2460 IF T>B(I) THEN 2470
2465 GOTO 2530
2470  I=I+F1
2480 GOTO 2020
2500 REM *** RIGHT ROUTINE
2510  T=F0
2515  D=F9
2517 PRINT #0
2520 PRINT #0,"A!F!C!G!A!F!C!G!A!F!C!G!";
2530  Q=PEEK(F4)
2535 IF Q<F5 THEN GOSUB 7000
2540 IF Q<>F5 THEN 2530
2545 IF Q<F5 THEN GOSUB 7000
2550  T=T+F1
2560 IF T>B(I) THEN 2570
2565 GOTO 2530
2570  I=I+F1
2580 GOTO 2020
```

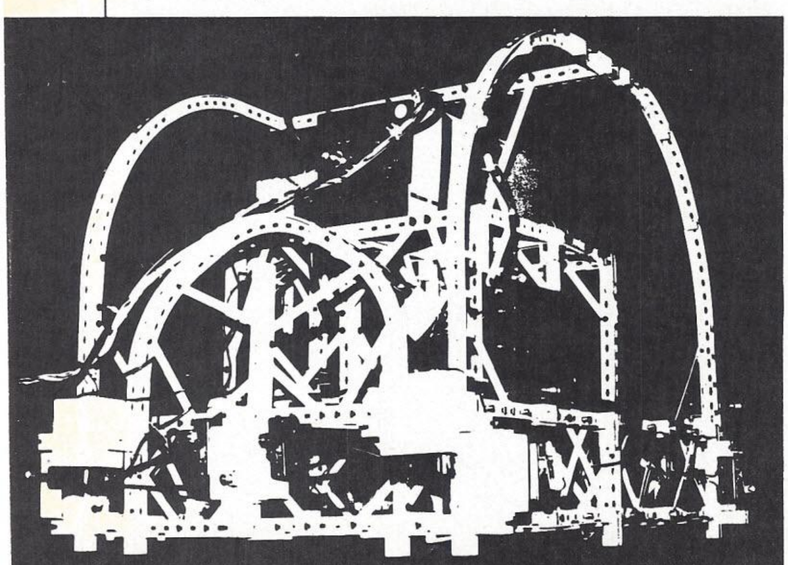
Program I - continued

```
2600 REM *** STOP ROUTINE
2610 PRINT #0,"D!H!";
2611 IF B(I)=0 THEN RETURN
2612 FOR Q6=1 TO B(I)*100
2613 NEXT Q6
2614  I=I+1
2620 GOTO 2020
2700 REM *** JUMP ROUTINE
2710  I=B(I)
2720 GOTO 2020
3000 PRINT #1," TURN TTY ON"
3010 INPUT " IS IT ON",G$
3020 IF LEFT$(G$,1)="Y" THEN 3040
3030 GOTO 3000
3040 PORT= P2
3050 INPUT " RANGE (1,R)",R
3060 PRINT "    TURN PUNCH ON"
3075 LINE= 240
3080 FOR I=1 TO20
3085 PRINT #P2,CHR$(255);
3090 NEXT I
3105 PRINT R
3107 FOR Q8=1 TO 10:PRINT CHR$(255);:NEXT Q8
3110 FOR I=1 TO R
3112 FOR Q8=1 TO 10:PRINT CHR$(255);:NEXT Q8
3115 PRINT A$(I);",";B(I)
3120 NEXT I
3130 FOR I=1 TO 20
3140 PRINT #P2,CHR$(255);
3150 NEXT I
3160 PORT= 1
3170 PRINT "SAVE DONE"
3180 RETURN
4000 PRINT "TURN TTY ON"
4010 INPUT "IS TTY ON",G$
4020 IF LEFT$(G$,1)<>"Y" THEN 4010
4030 PORT= P2
4040 PRINT "  PUT TAPE IN READER"
4050 PRINT "  TURN READER ON"
4060 INPUT R
4090 FOR I=1 TO R
4100 INPUT A$(I),B(I)
4110 NEXT I
4120 PORT= 1
4125 PRINT "  LOAD FINISHED"
4130 RETURN
5000 END
6000 PRINT #1,CHR$(16);CHR$(22);"LIST"
6010 PRINT #1
6020 INPUT "RANGE (R1,R2)",R1,R2
6030 FOR I=R1TOR2
6040 PRINT #P2,"   STEP #";I;"
     ";A$(I);"    ";B(I)
6050 NEXT I
6060 PRINT #1," LIST DONE"
6070 RETURN
7000 PRINT #0,"D!H!";
7002 FOR D9=1 TO 100
7003 NEXT D9
7010 ON D GOSUB 7300,7200,7500,7400
7020 FOR G4=1 TO 200
7030 NEXT G4
7095  T=B(I)
7100 RETURN
7200 PRINT #0,"A!E!C!G!A!E!C!G!A!E!C!G!";
7210 RETURN
7300 PRINT #0,"B!F!C!G!B!F!C!G!B!F!C!G!";
7310 RETURN
7400 PRINT #0,"B!E!C!G!";
7410 RETURN
7500 PRINT #0,"A!F!C!G!";
7510 RETURN
9900 PRINT PEEK(32790)
9910 GOTO 9900
```

Comments on Program I

Program Line 1–9 - Initialize constants, DIM statement. 32790 in Statement 1 is the sensor input location in decimal notation.

10–100 - This clears the CT-1024 terminal's screen and prints an options menu. Then it requests your choice. Line 78 sets the Teletype™ to Port 6. Line 90 sets the control terminal to the SWTPC CT-1024.



105–130 - This section checks the input for error and jumps to the appropriate routine through line 120.

1000–1100 - This section allows the user to input a program, a step at a time. See the command chart. When "#" is entered, the options menu is again displayed.

2000–2100 - This section runs the stored program. The robot performs each programmed step in order. Line 1085 prints out an error message if the command is invalid. Execution continues.

2200–2280 - This routine drives the robot forward B(I) spaces. It counts the pulses coming from the distance sensors, continually checking for any microswitch sensor that has reached an obstruction.

"T" is first set to 0 in line 2210. "T" is the counter of pulses from the distance sensor. "D" is then set to 1. "D" denotes the current direction. Line 2217 prints a carriage return/line feed to the relay interface port which resets the current character location points in BASIC. This avoids insertion of unwanted CR&LF's. Line 2220 activates the motors so that the robot goes forward. Line 2230 inputs the sensors into variable "Q". Line 2235 checks to see if a microswitch is closed indicating an obstruction. Line 2240 checks to see if a distance pulse is present. If not the program goes to Line 2230 and again inputs the sensors. Line 2245 check for sensor closure. Line 2250 increments the

distance pulse count. Line 2260 compares the new total to the prescribed amount. This checks the forward movement. Line 2265 re-reads sensors at line 2230 if the count has not reached the programmed distance. Line 2270 increments the step counter. Line 2280 returns to command decoder (Line 2020).
2300–2380 - The backward routine – duplicates lines 2200–2280 although line 2320 is different.
2400–2480 - The left routine – same as above.
2500–2580 - The right routine – same as above.
2600–2620 - The stop routine – if B(I)=0, return to options menu. If not, wait B(I) time periods.
2700–2720 - Jump routine – go to control statement #B(I).
3000–3180 - This routine saves a control program on paper tape. Line 3107 prints nulls which are necessary for timing. Line 3112 does the same thing. Linc 3115 outputs a set of values.
4000–4130 - The LOAD routine loads a paper tape generated by the SAVE routine. Line 4060 gets the number of steps to come from the tape.
6000–6070 - The LIST routine is used to list the stored program.
7000–7510 - This routine is called whenever a sensor closure is detected. The routine stops the robot and waits for a while, (lines 7000-7003). Then Line 7010 directs the program to a line which makes the robot motors avoid the obstruction that has been detected. "D" is a variable whose value can be 1, 2, 3, or 4. 1 means forward; 2, back; 3, left; 4, right. Line 7010 thus selects proper directions for the robot. Lines 7020–7030 cause a fixed delay. Line 7095 should be "T=B(I)". This, in effect, causes the robot to skip the step involving the obstruction.
9900–9910 - A utility routine for printing sensor inputs in decimal on the terminal.

Program II

```
0001 GOSUB 5000
0002  P2=3
0005 GOSUB 100
0010  A=PEEK(32782)
0015 IF A>62 THEN 10
0016  B=PEEK(32782)
0017 IF B<129THEN16
0030 GOSUB 1000
0035 GOSUB 500
0038 GOSUB 8000
0040 GOTO 10
0100 FOR H=0 TO 63
0110 FOR V=128 TO 223
0120 PRINT #P2
0130 PRINT #P2,CHR$(H);CHR$(V);
0140 NEXT V
0150 NEXT H
0155 RETURN
0500 REM
0525 PRINT #P2,CHR$(A);CHR$(B);
0540 RETURN
1000 PRINT #P2,CHR$(A+64);CHR$(B);
1035 GOSUB 500
1040 RETURN
1050 GOTO 10
5000 PRINT #1,CHR$(16);CHR$(22);
5010 PRINT #1,"                    F"
5020 PRINT #1:PRINT #1:PRINT #1
5030 PRINT #1,"          ***************"
5040 FOR Z9=1 TO 6
5050 PRINT #1, "L          *
     *          R"
5060 NEXT Z9
5070 PRINT #1,"          ***************"
5080 PRINT #1:PRINT #1:PRINT #1:PRINT #1,"
     B";
5095 PRINT #1,CHR$(16);
5100 RETURN
8000 IF A>45 THEN 8050
8005 IF A<18 THEN 8050
8010 IF B>145 THEN 8050
8015 IF B<120 THEN 8050
8020 REM *** FORWARD MOVEMENT
8025 GOSUB 8500:PRINT #0,"A!E!C!G!";:RETURN
8050 IF A>45 THEN 0
8055 IF A <18 THEN 8100
8060 IF B<190 THEN 8100
8065 REM *** BACKWARDS MOVEMENT
8070 GOSUB 8500 :PRINT #0,"B!F!C!G!";:RETURN
8100 IF A>10 THEN 8150
8105 IF B<160 THEN 8150
8110 IF B>180 THEN 8150
8115 REM *** LEFT HARD TURN
8120 GOSUB 8500:PRINT #0,"A!F!C!G!";:RETURN
8150 IF A<55 THEN 8200
8155 IF B<160 THEN 8200
8159 IF B>180 THEN 8200
8165 REM *** RIGHT HARD TURN
8170 GOSUB 8500:PRINT #0,"B!E!C!G!";:RETURN
8200 REM *** STOP
8205 PRINT #0,"D!H!";:PRINT #1,"";:RETURN
8500 PRINT "";
8505 RETURN
```

Comments on Program II

This program uses the SWTPC Joystick and SWTPC GT-6144 Graphics Display. A map (see diagram) is put on the video screen showing front, back, right and left. A blinking cursor moves around the screen as the Joystick is moved. When the cursor moves near one of the letters F, B, L, or R on the screen, the robot moves in that direction. The screen now changes from computer-generated images to video signals from the robot's camera. When the stick is again centered, the robot stops, and the map reappears.

To understand the comments on the program which follows, you must understand the two peripheral devices.

The Joystick is interfaced to the input of a SWTPC parallel I/O port. The stick moves in two dimensions. The onboard analog to digital converter transforms the stick motion into output values. "A" is the horizontal stick position value, and "B" is the vertical position value. Joystick continually outputs "ABABABABAB – " at a rate of fifty times a second. "A" varies from 1 to 62 (decimal) while "B" varies from 218 to 129. Thus, A=1, B=129, would be the lower left corner; A=62, B=129, would be the lower right corner; A=62, B=218, would be the upper right corner; and A=1 B=218, would be the upper left corner.

The GT-6144 Graphics Device displays rectangular dots on the video screen. These dots overlay at the same time, as characters from the CT-1024. The graphics display is a grid

of 63 horizontal dots across and 95 vertical dots, for a total of 6,144 dots. The display connects to the output side of the same parallel input/output port that controls the Joystick.

Any dot can be turned on and off, one at a time, as follows: The first byte sent to the board must be a horizontal position $(0\text{-}63_{10})$ as well as the command to write or erase a dot if necessary. The most significant bit, B_7, is set to 0 to indicate a horizontal coordinate. B6 is set to 1 to write or 0 to erase a rectangle.

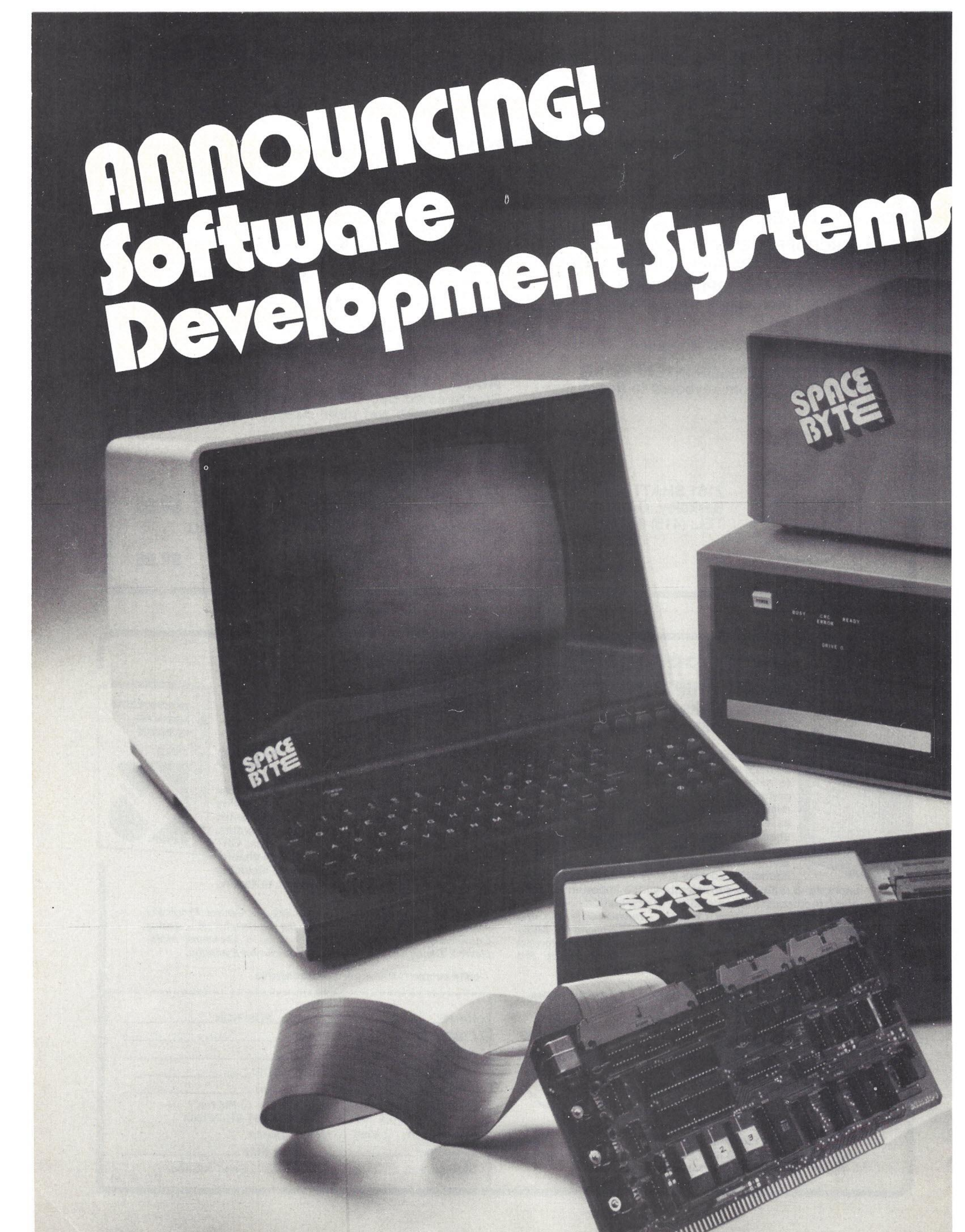
Data bits $B_5 - B_0$ contain the horizontal position which varies from 0 to 63 (decimal).

The second byte is the vertical position. Bit B_7 must be set to 1 to indicate a vertical coordinate. Bits $B_6 - B_0$ contain vertical coordinates 0 to 95 (decimal).

Several special functions are also available. Sending a control character using the CHRs Function accomplishes the following actions:

CODE (decimal)	RESULT
224	Invert screen
225	Enable graphics (GT-6144)
226	Disable CT-1024
227	Not used
228	Normal screen
229	Disable graphics (GT-6144)
230	Enable CT-1024
231	Not used

Program Line 1 - Write the map on the screen.
2 - Set port of Graphics/Joystick.
5 - Erase the graphics, 1 dot at a time.
10–17 - These lines produce updated horizontal and vertical coordinates. Peek (32782) returns a decimal value from the Joystick. The program receives an "A" value, which varies from 1 to 62. Then it receives the "B" value, which varies from 129 to 328.
30 - Display the cursor at the coordinate (A,B).
35 - Erase the cursor at the coordinate (A,B).
38 - Determine what to do as a result of where the cursor is (A,B).
40 - Loop back and get a new value of (A,B).
100-155 - Erase the graphics display (subroutine).
500–540 - Erase a dot at A,B. (Subroutine).
1000–1040 - Write a dot at A,B, then erase it. (Subroutine).
5000–5100 - This subroutine writes the map on the screen.
8000–8015 - This code compares (A,B) with the range of values for a forward move. If any test fails, the program goes to 8050 and starts checking for a reverse motion. If (A,B) falls within the specified range then a forward motor command is sent to the relay interface. The program then goes back for a new ((A,B) value.
8050–8060 - These lines test for a backward motion, similar to Lines 8000–8015.
8025 - Change screen from computer to video image. Set robot going forward.
8070 - Similar to 8025 except backwards.
8100–8110 - Check for left turn.
8120 - Similar to 8025 except left.
8150–8159 - Check for right movement.
8170 - Similar to 8025 except right.
8200–8205 - If all tests fail, stop motors, restore computer image to screen, go back for new (A,B).
8500–8505 - Subroutine to change screen from computer image to video image. PC


ANNOUNCING!
Software
Development Systems
SPACE BYTE
SPACE BYTE
SPACE BYTE
POWER
BUSY
CRC ERROR
READY
DRIVE 0


by SPACE BYTE™
SPACE BYTE Software Development Systems are versatile and cost effective; featuring the SPACE BYTE 8085 CPU, self contained computer, 16K SPACE BYTE fully static RAM and the new SPACE BYTE 2708/2716 EPROM Programmer. All PROM programming, I/O handling FDOS, and complete debugging routines, with breakpoints are intrinsic to the SPACE BYTE system monitor. Complete software support too, with iCOM FDOS III, 8080, 8085, Z-80 relocatable macro assembler, operating system, Disk Extended Basic, CP/M and FORTRAN-80 and the SPACE BYTE exclusive 3870/F8 cross assembler.
THE SPACE BYTE SYSTEMS GROUP
Software Development Systems
Business Systems
Word Processors
SPACE BYTE MICRO COMPUTER PRODUCTS ARE SUPPORTED BY COMPUTER RETAILERS WORLD WIDE.
1720 Pontius Ave. Suite 201
Los Angeles, CA 90025
SPACE BYTE SOFTWARE DEVELOPMENT SYSTEM PRICING STARTS AT $5900.
CPU • RAM
EPROM
PROGRAMMER
Special Offer
Save $150.00
Present the special discount coupon (found later in this magazine) to any participating computer retailer, and they will package the system for your particular application. YOU SAVE 10% of the end user price on the combination SPACE BYTE 8085 CPU, 16K SPACE BYTE and 2708/2716 EPROM Programmer (total price $1347.). Offer good at participating computer retailers only, void where prohibited, taxed, or restricted by law. Offer expires March 31, 1978.
CIRCLE 18

This is the House

Somewhere in your city an entrepreneur wants a computer's help. This entrepreneur knows he needs complete, timely information to succeed. If the Lemonade Computer Company can help him make decisions based on knowledge rather than intuition, he will be happy to reimburse the company for its services.

This entrepreneur (let's call him Joe) is in an unusual business – housing resales. Joe starts out by buying a run-down house, a "fixer-upper", located in a good neighborhood at a price 10% or more below the current market. He then repairs and remodels the house and re-sells the property at the true market value as soon as possible. Joe makes money when the difference between his buying and selling prices exceeds his total expenses. He needs complete, accurate data when he purchases the property and when he negotiates its sale. In addition, he must schedule the repair work efficiently. Here is where the Lemonade Estimating and Scheduling Service (LESS) offered by the Lemonade Computer Company can be of value.

The central idea of LESS is to estimate expenses by predefined categories and use these to calcualte the break-even resale price for the property. After the sale, comparing actual expenses to the estimates helps sharpen future estimates. From the actuals, he also calculates the profit on the transaction. Expense categories vary by entrepreneur and need customizing for each individual (see representative list in the LESS Report 1 – Estimates.)

LESS is organized conceptually in three stages; Pre-buy, Repair, and Post-sale. Each stage is described below in terms of its input, output and purpose. The development and implementation of the actual programs is left as a non-trivial task for the reader.

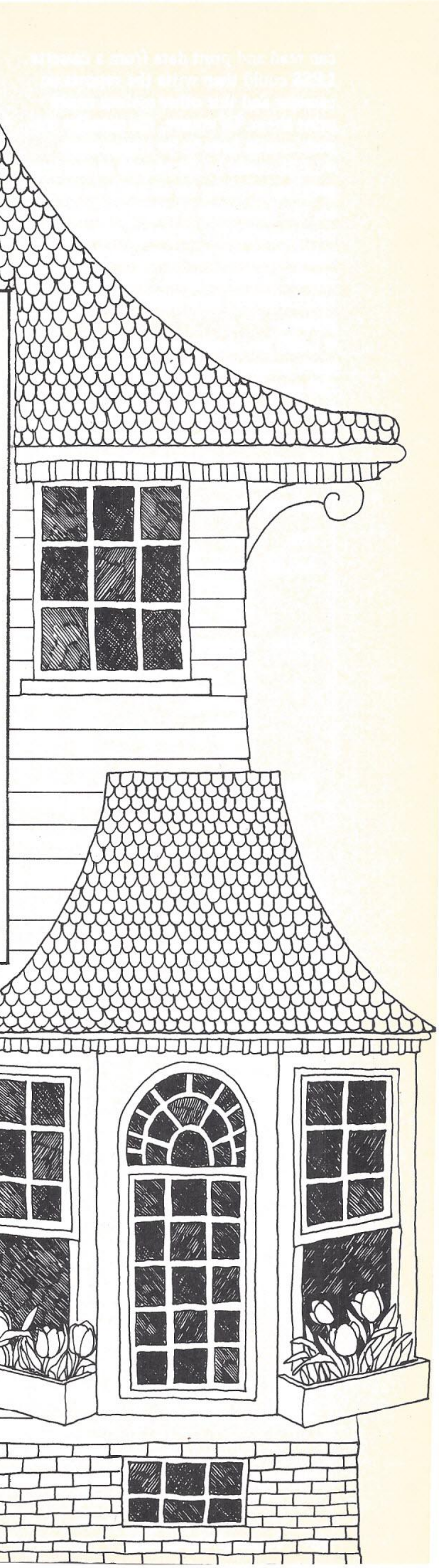
Pre-buy

This stage assists the customer in deciding the highest price he can pay for the house and still make a reasonable profit. The basic input is (a) the estimated expense of repairing the house, by category, (b) the estimated expense associated with re-sale of the property and (c) the proposed cost of the house itself. All input is collected according to the predetermined expense category Input item (c) probably varies during successive runs of the program, and may automatically cause other expenses – points, escrow fees – to change. LESS Report 1, the output of the Pre-buy phase, gives our entrepreneur the price at which the house must be sold to make money.

Repair

The Repair phase is really a scheduling program. Each category of expense established by Joe may represent a distinct item of work to be done on the house. Each of these items will take a definite amount of time (e.g., one day for carpeting). Repairs should also be done in a definite order, e.g., painting before carpeting. The scheduling procedure itself will depend on the people Joe uses to do the work and his past experience with them.

Estimates developed during Phase 1 (the Pre-buy) provide the input for the Repair phase. In this phase LESS then schedules the work according to whether that item of repair is called for (has a non-zero estimate). In addition, this phase computes the minimum number of days required before the house can be shown. This estimate helps the entrepreneur set up his open



that LESS Built!

BY REGINALD GATES

house. LESS Report 2 shows the repair schedule and elapsed days computation – the LESS Scheduling Report.

Post-sale

After the property is sold, this phase of LESS helps the entrepreneur determine profit or loss on the transaction by accumulating actual expenses by category.

Several items are input: (a) actual cost and actual selling price of the house, (b) number of house payments made by Joe while the property remained vacant, (c) actual expenses associated with the transaction and (d) estimates of costs during Phase 1. The output report develops a cost variance by comparing actual expense with estimated expense, and also figures the percentage of total cost represented by actual expense. Joe uses a simple formula to calculate profit: profit = selling price - costs; where costs = purchase price + repair costs + house payments + sale expenses. This report forms a valuable part of the entrepreneur's financial records. LESS Report 3 – Actuals shows how the data looks .

Development of LESS

As mentioned before, the design, coding and testing of the program (or programs) for LESS are not as easy as they first sound. Since actual design differs with system resources available, only a few general guidelines apply.

- The reports produced by the system almost demand that the Lemonade Computer Company have some sort of hard-copy device available. If the company doesn't own one, perhaps they could locate a club or individual who

LESS REPORT 1

TRANSACTION ID XXXXX ADDRESS X------------------X

CAT	DESCRIPTION	ESTIMATE	PER
0	COST OF HOUSE	45,000	-
1	CARPETING	500	34
2	DRAPES	150	10
3	ELECTRICAL WORK	0	
4	PAINTING - INT	50	03
5	PAINTING - EXT	225	15
6	WINDOW REPAIR	75	05
7	HOUSE CLEANING	35	02
9	CARPENTRY	0	
10	NEW APPLIANCES	325	22
11	PLUMBING REPAIR	75	05
12	OTHER - XXXXXXXXXXX	0	
TOTAL	1 THRU 12	1,470	22
13	POINTS	4,400	82
14	TITLE SEARCH	35	01
15	INSURANCE/IMPOUNDS	425	08
16	TERMITE INSP	55	01
17	OTHER ESCROW FEES	400	07
TOTAL	13 THRU 17	5,315	78
TOTAL	1 THRU 17	6,785	-
GRAND	TOTAL 0 THRU 17	51,785	-

can read and print data from a cassette. LESS could then write the reports on cassette and this other system could print them. Of course, the cassette formats would have to be compatible.

- LESS could be developed as one large program working on an array in core. If so, the Lemonade Computer Company must provide for retaining the array values between execution of the various phases of the program. You also need mechanism to pass control between phases.
- LESS could also operate as three separate programs, one for each phase. In this case, the provision system, for transfering must make estimate data

LESS REPORT 2

TRANSACTION ID XXXXX

TYPE	ITEM	MIN	MAX
OUTSIDE	WINDOW REPAIR	01	02
	PAINTING - EXT	03	07
	YARD WORK	01	01
	TOTAL -------------	05	10
INSIDE	PLUMBING	01	02
	(PLUMBING INSP)	02	05
	NEW APPLIANCES	01	03
	PAINTING - INT	01	02
	CARPETING	01	02
	DRAPES	01	03
	HOUSE CLEANING	01	01
	TOTAL -------------	08	18
GRAND TOTAL ---------------		13	28

between phases (floppy disks anyone?). This approach will take less core.

- In either approach to LESS's design, the plan must permit customizing the categories for the entrepreneur's requirements.
- Regardless of the development path chosen, LESS should be completely tested using a reasonable set of categories *before* attempting to locate a nearby entrepreneur.

The data processing industry is filled with examples of technically excellent products that were commercial failures because of marketing problems. Pricing and marketing know-how will separate successful Lemonade Computer Companies from the rest. Even with a working model of LESS, the job is only half done.

One of the first questions any prospective client will ask is "How much will this cost?". To price this service the Lemonade Computer Company will need to answer some basic questions about itself.

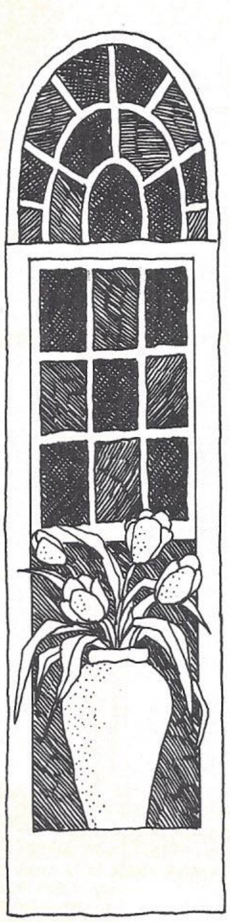
First, is the company in business to make a profit? If the Lemonade Company intends to make a profit, its services must be priced to recover equipment cost over a reasonable period and repay the programmer for time spent developing the services. On the other hand, if the hardware was purchased primarily for personal enjoyment, and if the programmer does not wish to be reimbursed, *any* revenue from the services is considered profit. In this mode, the price of the company's offerings can be much lower. Most Lemonade Computer Companies will fall into the latter category.

The next question is whether to price the service on a fixed monthly fee basis, or on a transaction basis. While a fixed monthly fee would assure a known income, charging the customer for each use of the service seems to be the most feasible approach. In the case of LESS, the user would pay a fixed amount for each report, since these are the major outputs of the system. This method would eliminate any problems with trying to measure "connect time". Subjectively, a price of $5 to $10 a report would seem about right.

Now that LESS has been priced according to the purpose and policies of the Lemonade Company, the next task is to locate prospective users. Talk to your friends, your realtor, escrow officers, people you know in the building trades, until you find a possible client. Contact businesses and explain what you're doing. If you must make the initial contact by mail, it would be a good idea to send along a sample of LESS's reports. If possible, invite the propsect to a demonstration of the system. You might even offer the service free for a trial period to whet interest. (Hopefully, at the end of the trial period your prospect will like LESS so much, or will have become so dependent on it, that he will continue.)

Enhancements

What we've described has been the "kernel" of the Lemonade Estimating and Scheduling Service. Further system additions beckon. Perhaps you could expand Phase 3 to give the client a year-to-date expense report for tax purposes. You could modify Phase 1 to accept expense input directly or calculate the amount, given certain parameters for a category. For example, given the yards of carpeting required, Phase 1 should be able to compute the cost. Could LESS also expand to generate Accounts Payable as a function of Phase 3? The possibilities are intriguing. LESS illustrates again that the personal computer can do other things than play Star Trek. The system fills a definite need in a real business. As such, it could be a legitimate commercial offering. While LESS alone would not support a Lemonade Computer Company, it could be a revenue-generating application.



LESS REPORT 3

TRANSACTION ID XXXXX ADDRESS X------------------X

CAT	DESCRIPTION	EST	ACTUAL	PER	VAR
0	COST OF HOUSE	45,000	46,200	87	1,200
1	CARPETING	500	550	01	50
2	DRAPES	150	130	00	-20
3	ELECTRICAL WORK	0	0		
4	PAINTING - INT	50	50	00	0
5	PAINTING - EXT	225	275	01	50
6	WINDOW REPAIR	75	35	00	-40
7	HOUSE CLEANING	35	35	00	0
8	YARD WORK	35	35	00	0
9	CARPENTRY	0	0		0
10	NEW APPLIANCES	325	350	01	25
11	PLUMBING REPAIR	75	150	00	75
12	OTHER	0	0		0
TOTAL	1 THRU 12 ----------	1,470	1,610	03	140
13	POINTS	4,400	4,400	08	0
14	TITLE SEARCH	35	35	00	0
15	INSURANCE/IMPOUNDS	425	125	00	-300
16	TERMITE INSP	55	55	00	0
17	OTHER ESCROW FEES	400	455	01	55
TOTAL	13 THRU 17 ---------	5,315	5,070	10	
TOTAL	1 THRU 17 ----------	6,785	6,680	13	-105
GRAND	TOTAL 0 THRU 17 ----	51,785	52,880		1,095
*****	SELLING PRICE *****	54,500			
*****	HOUSE PAYMENTS ****	850			
*****	PROFIT ****	770			

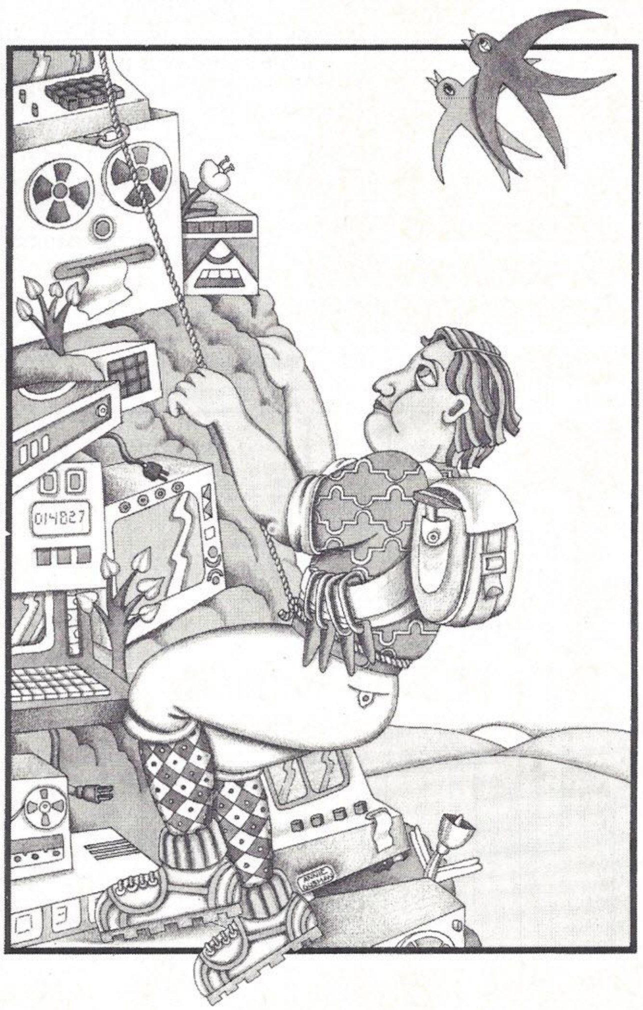
Scaling the Cliffs to Computer Mastery

Part II

BY RAYMOND HOWELL

In Part I of this article, (January '78) you learned enough about microcomputers to get your hands on one and start using it. But several sticky questions might arise where the answers will not be found in manuals or magazine articles at this stage of your knowedge. A closer look at a few basic areas should prove helpful in your progress through the world of microcomputers.

Up to now I've used simple examples and terminology. I'll begin, now, to use specific terminology more consistent with what you will find in the microcomputer world.



Octal and hexadecimal numbering systems

In our example with I/O ports we used eight lights to tell us what was on the data bus, 16 lights to indicate the address that was on the address bus, and 16 switches to enter the data into the computer. We had no choice but to talk to the computer in its own language – machine language – which uses the binary number system. If you're like me though, numbers above decimal 15 become rather difficult to handle.

We're now familiar with two number systems: the decimal number system (base ten) that uses ten numbers, 0-9; and the binary system (base two) that uses only two numbers, 0 and 1. But there are other useful systems using different number bases. Consider these two systems – octal and hexadecimal.

The octal number system (base eight) uses eight numbers, 0-7; while hexadecimal (base 16) uses 16 numbers, 0-F.

Base eight has no number 8 or 9. Rather, when we reach number 7 we start over with 10 as in the decimal system (after 9 comes 10).

The hexadecimal system (hex means 6 and decimal means 10, thus 6 plus 10 equals base 16), proposes a different story. Here we use 16 numbers. The first ten numbers are familiar, 0-9, but what about the other six numbers? Well, we use the first six letters of the alphabet, A-F.

In the first part of this article we considered binary numbers of eight digits, a byte. Now look at a byte of data similar to those that might appear on the eight switches on the front panel of your sample computer. It's easier to use two digits rather than eight digits to represent a given quantity.

In Fig 1 a-c we see eight digit binary numbers and the same quantity expressed with two hex (short for hexadecimal) digits. The eight digit binary number is divided into two 4-digit binary numbers and each four digit binary is assigned its equivalent hex number. Hex is much easier to work with when writing or transferring data with any degree of accuracy.

The highest single digit hex number, F, corresponds to 1111 in binary. It's also the largest number that can be expressed with four binary digits. This is why it's convenient to use two hex digits to represent eight binary digits.

The largest single octal digit, 7. (Fig 1 d-f) corresponds to the largest three digit binary number 111. So by starting on the right side of an eight digit binary number and dividing it into groups of three digits, you easily convert a byte of data from binary to octal. The left most group of the eight digit binary number will contain only two binary digits.

Most manufacturers of CPU's publish instruction sets with operator codes expressed in hexadecimal. Some instruction sets give operator codes in hexadecimal and octal. Binary is also sometimes included for the operator code. This is useful only if your I/O input has eight data switches. If you're using a keyboard input and a teletype or video output, you'll most likely use hex to talk to your computer.

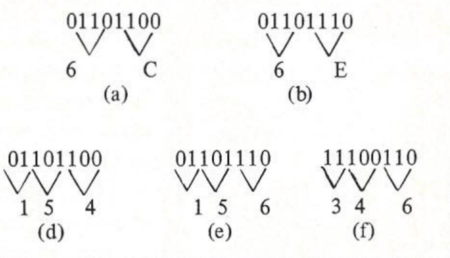
Bytes and words

Another term you will encounter in computer language is "word". We talked about bits and bytes, and in our sample

Illustration by Annie Gusman

microcomputer the CPU worked with one byte of data at a time. But in some minicomputers, the CPU, memory and data bus work with 16 bits or two bytes of data at a time. Whatever the computer handles, 8 bits or 16 bits at a time, it is called a "word". Most microcomputers though, work with 8 bit words – a byte. Some larger computers work with 32 bit or four byte words. Although a byte and a word are synonymous in most microcomputers you shouldn't confuse them.

Figure 1



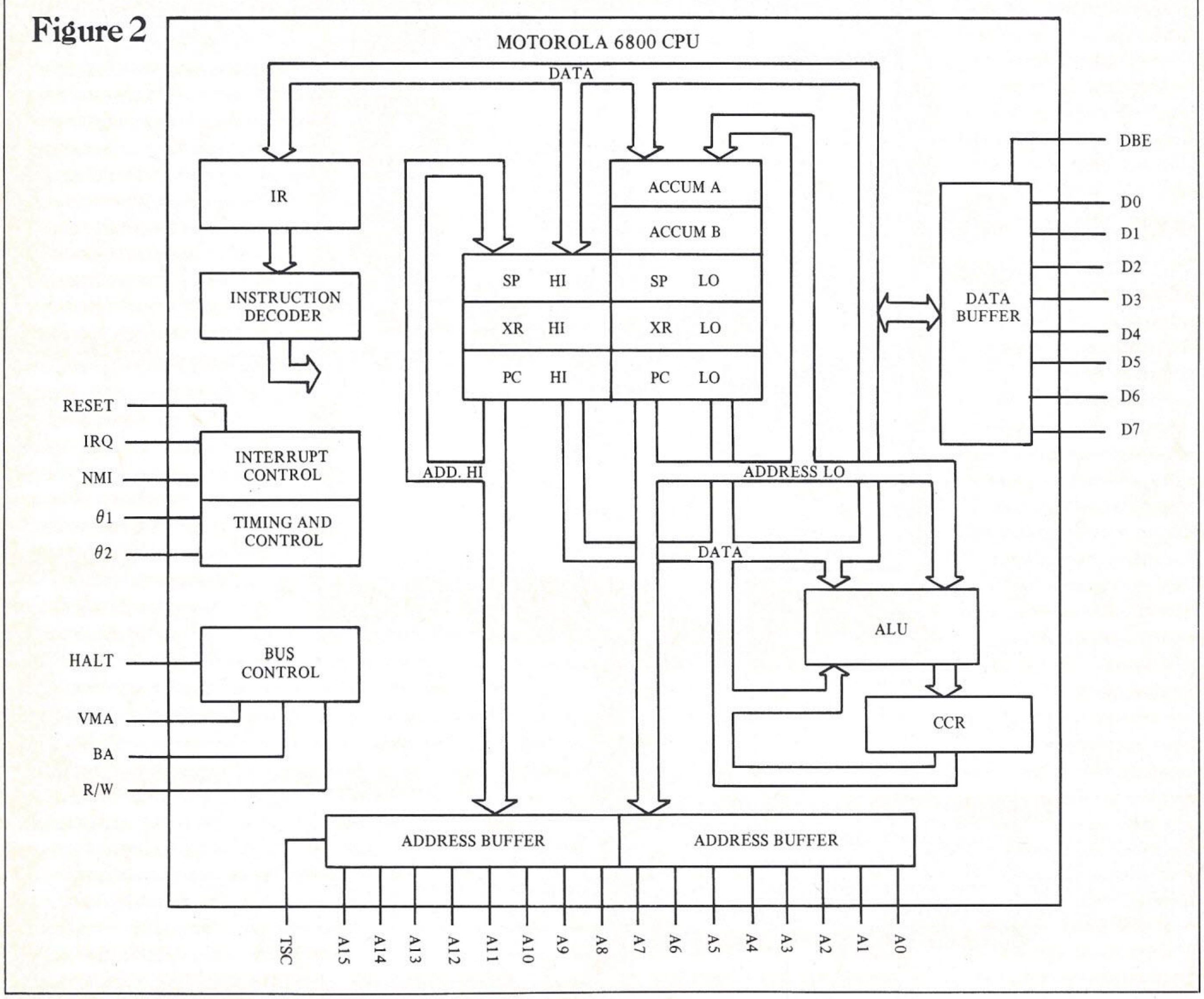
Types of memories

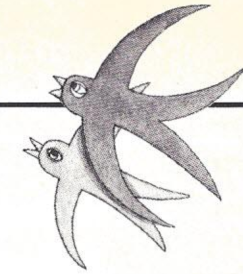
Memories can be categorized in several different ways. One way is volatile and non-volatile. Volatile memory will not retain data when power is removed while non-volatile memory does not require external power to maintain its contents. Some volatile memory comes with a back-up power supply that operates from batteries.

Not all memory can store data from the CPU. The type of memory that cannot is called read-only memory (ROM). So here is another way to categorize memory: ROM and RAM (or read-and-write memory). ROM and RAM can be read from, but only RAM can be written into by the CPU. ROM memory is programmed during manufacturing, and contains a program that will be used over and over. The program remains permanently in the computer and cannot be erased or written on top of.

Along with RAM and ROM you run across terms like PROM, EPROM and others. PROM (programmable read-only memory) is programmed in the field by the user rather

Figure 2





than at the time the machine is manufactured. Once PROM has data stored it really becomes ROM because its data cannot be changed. EPROM (erasable programmable read-only memory) is similar to PROM except that if you make a mistake in programming it or want to change the data, you can erase and reprogram the EPROM.

Software and hardware

We've talked about CPU's, memory, buses and I/O's. All of these things we can touch and see, and they're called hardware. Software we cannot see. Software is the program, stored in memory, that causes the computer to perform a given task. We see the software only when it's printed out.

When you write a program into ROM, PROM or EPROM so that it cannot be altered by the computer, you call it firmware. Software documentation is the written copy of the software or firmware.

If you start looking at actual instruction sets of various CPU's, you'll find each instruction associated with a term such as "clock cycles" or "machine cycles". This refers to the number of cycles the CPU clock has to make in order to perform that particular instruction. This information becomes important when you program your computer to perform timed events.

Expanded CPU

In our previous study of the CPU we did not discuss three important items. The first is the Arithmetic Logic Unit (ALU). Previously we considered the accumulator responsible for all the math. In reality, the accumulator receives data from the data bus. But if you require an arithmetic or logic (OR, AND, NOR) operation, the ALU performs that task (see Fig 2). If the accumulator receives data and an instruction to store that data in another location, then the accumulator performs that operation without involving the ALU. If the instruction were ADD, or any other arithmetic or logic instruction, the ALU would perform the math and leave the results in the Accumulator.

Another new item, the Condition Code Register (CCR), connects to the ALU and to the data bus. The CCR records some results of the ALU action such as: did the operation performed by the ALU result in zero in the accumulator? did the ALU operation create a carry, that is, a number greater than eight binary digits? was the result positive or negative? These conditions are sometimes spoken of as flags – zero flag, carry flag, sign flag. Each CPU manufacturer has various kinds and numbers of flags.

The CCR also connects to the data bus so that its contents can be read the same as any memory location. When you look at different kinds of instructions, you can see the importance of the CCR.

Mnemonics

If you've been looking at actual instruction sets, you've probably come across the term mnemonic. A mnemonic is an abbreviation for the real meaning of the instruction. For example, the mnemonic ADDM might mean "add the contents of a memory location to the accumulator contents and leave the result in the accumulator." The use of mnemonics makes for briefer programs while enhancing clarity. Note that the instruction set gives the operator code and the number of operands that must accompany the operator code to make the instruction complete. Some instructions do not have an operand, while others have one operand and still others have two operands.

Instruction sets

The instruction set for any microcomputer can be arranged in five groups; data transfer, arithmetic, logical, branch and control. The data transfer group of the instruction set includes instructions that are used to move data between memory and the CPU or between I/O ports and the CPU.

The arithmetic group of instructions performs addition, subtraction, incrementing and decrementing. These operations take place in various registers as well as memory. The logic group of instructions perform logic operations such as AND, OR, EXCLUSIVE OR, compare, rotate, and complement in registers or memory.

The branch instruction group causes the program counter to jump to another location in memory and begin executing the program at the location. These instructions are either conditional or unconditional. An unconditional branch instruction, sometimes called jump or skip, will tell the CPU to move to a given location and execute the program there. A conditional branch depends on the last operation of the CPU and how that operation affected the CCR. For example, "branch if zero flag is set" will instruct the CPU to examine the CCR to see if the zero flag is set. The CPU will make a decision based on its findings. If the zero flag is set, the program counter will jump to the location pointed to by the operand of the branch instruction. If the zero flag is not set, the program counter will bypass the branch instruction and execution will continue at the next command.

The branch group of instructions also includes return or non-return branches. With a return branch instruction the CPU stores the address of the branch instruction in some memory location. The program counter will jump to the location pointed to by the branch instruction and the CPU ex-

Manufacturers do not always break down their instruction sets into clear groups, but if you can visualize the five categories, it can help as you learn a particular CPU's instruction set and begin to write your own program.

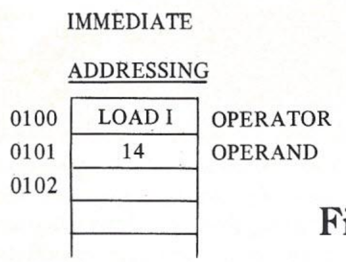


Figure 3

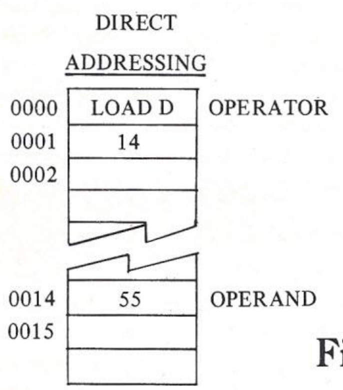


Figure 4

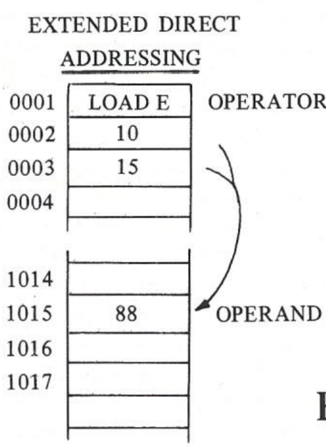


Figure 5

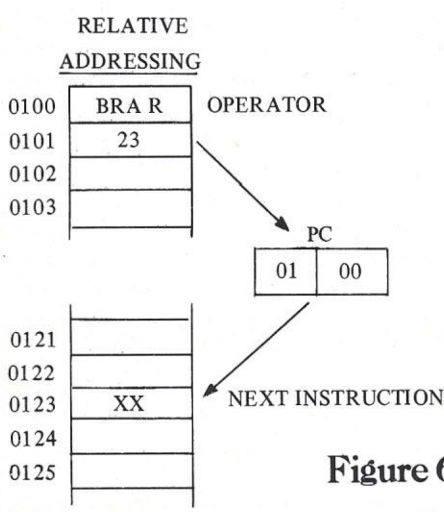


Figure 6

ecutes that program until it comes to a return instruction. At that point the program counter will get the previously saved branch instruction address and return to the main program and continue executing from where it left off. With non-return branch the CPU does *not* store the address of the branch instruction and thus it does not return to that program.

The control group of instructions manipulate the program counter, control register and conditional code register. These instructions may change the contents of these registers, increment them or decrement them.

Manufacturers do not always break down their instruction sets into clear groups, but if you can visualize these five categories, it can help as you learn a particular CPU's instruction set and begin to write your own programs.

Addressing modes

Another important aspect of an instruction set concerns addressing modes. Up to now we have spoken of an instruction as being composed of two parts, the operator and the operand. The operator is the basic instruction such as LOAD or ADD; the operands answers the question "what" or "where". Thus, LOAD the accumulator with "what" would mean to load with the operand. LOAD the accumulator with "where" means the operand points to the memory location where the data is stored that is to be loaded into the accumulator.

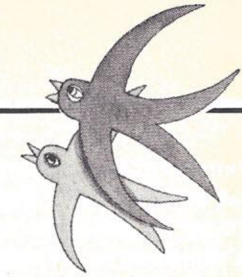
I must confess that I'm using the word "operand" incorrectly, but only because I haven't found another term to explain the two parts of an instruction. In reality the operand is data that is to be operated on. The LOAD instruction may operate on the byte of data following its operator in which case that byte contains the operand. Or the LOAD instruction may look at the byte of data following the operator to find out where the operand is stored in memory.

This brings us to addressing modes. Addressing modes refer to the different ways the CPU locates the operand or data to be operated on. In the simplest addressing mode, the implied or inherent mode, the address is implied. For example, the instruction CLEAR implies that the data to be operated on is in the accumulator. Instructions with addresses implied are one byte instructions.

Another addressing mode, the immediate mode, means that the second byte of the instruction contains the operand (see Fig 3). When this instruction has been executed, the accumulator will contain the number 14.

A direct addressing mode means that the second byte of the instruction contains the memory location of the operand (see Fig 4). At the completion of the LOAD instruction (in Fig 4), the accumulator will contain the number 55. Quite often the direct mode of addressing requires that the instruction consist of three bytes, the first being the operator, the second byte containing the high byte of the address and the third byte containing the low byte of the address; the instruction can thus address any location from 0000 to $FFFF_{16}$ using a 16 bit binary address (see Fig 5). At the completion of this instruction, the accumulator will contain the number 88.

You use the relative mode of addressing with the branch (conditional or unconditional) instructions. By adding the



content of the second byte of the branch instruction to the content of the program counter (which contains the address of the operator of that branch instruction) you end up with the address of the next instruction that the CPU will execute in the program counter. For example, in Fig 6 the present instruction being executed is at location 0100, thus the program counter contains 0100. The second byte of the branch instruction is 23 which is added to the 0100 in the program counter resulting in 0123. In relative addressing, the second byte of the instruction does not point to the data that is to be operated on, but rather it points to another instruction. With relative addressing there are limits to the distance you can branch from the current instruction since only one byte specifies that distance.

One 8-bit binary number can represent up to 256_{10} memory locations and CPU's are designed to branch forward and backward. This means you can reach an address within +/- 128_{10} locations of the branch instruction using the relative mode of addressing.

We know that microcomputers communicate with the outside world via input and output ports called I/O's. When the CPU writes data into a memory location, it does it in parallel format. That is, all eight bits are handled simultaneously via eight parallel conductors, one conductor for each bit of data in a byte. Since the entire internal operation of the microcomputer is parallel as far as the computer is concerned, parallel I/Os are the logical way to communicate. But much of the equipment you may connect to your computer, such as a teletype machine or a tape recorder, requires serial data transfer. A single conductor sends serial data so it must be sent one bit at a time. In order for the computer to do this it must serialize its data. This can be accomplished either by hardware or software. Many IC's are manufactured that take parallel data and convert it to serial and vice-versa. But you can perform the same task using a parallel I/O and software to serialize the data through only one of the parallel ports.

Computer language

Computers understand "machine language" and only machine language – nothing more than binary 1's and 0's. Instruction sets list their operator codes in machine languate. Often these operator codes appear in hexadecimal, but if you're talking to your computer in hex, there has to be an interface device between you and the CPU to translate the hex number to binary. But, whether you're using hexadecimal, octal or binary, you're still using machine language.

You can also use "assembly language" with computers. To talk to your computer in assembly language, you use mnemonics. Instead of giving an operator code to the computer, use a mnemonic such as LOAD. For the CPU to understand your instruction, the computer must contain a program that converts the mnemonic to machine language or 1's and 0's. Assembly language is easier than machine language because mnemonics are more easily remembered than the operator code for that mnemonic.

Higher level languages such as BASIC, COBAL and FORTRAN go much further than assembly language because they enable you to talk to your computer in plain English. Well, almost plain English. You have to use the proper choice of words in the correct format. To function, the computer must contain a program for interpreting what you say.

An advantage when using machine language is that you do not have to invest in the extra memory needed to store higher level language programs. Another advantage is that you'll learn more about your computer by communicating with it in its own language, and you'll be able to watch how the computer goes about accomplishing a given task. One of the most rewarding aspects of microcomputers is conceiving a task and devising a way to perform it using the instruction set and any hardware you desire. A game you can play with no holds barred! Yet the rules (instruction set) are fixed and the computer will not let you break even one of them.

Programming

Several basic microcomputers are available on single printed circuit boards. With a suitable power supply and a teletype machine connected to the computer board, you're in business. You type a command in machine language and the computer responds by typing a reply in machine language (usually in hexadecimal). This type of communication is possible because on that one printed circuit board along with the CPU chip and some RAM, there's a ROM that contains a monitor program. This program allows you to exam-

One of the most rewarding aspects of microcomputers is conceiving a task and devising a way to perform it using the instruction set and any hardware you desire. A game you can play with no holds barred! Yet the rules (instruction set) are fixed and the computer will not let you break even one of them.

ine the contents of any memory location, change the contents of any location in RAM, set the program counter to any desired location and execute a program you've written into a block of memory. Some microcomputer systems have monitors that call other programs stored on ROM, such as a program to read and write on cassette tape.

More elaborate microcomputers usually contain more firmware. Each system though has manuals for firmware documentation and instructions. Again, only your imagination will limit what you get out of the available firmware.

To program your computer in machine language to do a particular task, try following these steps: define the task; flow chart the process; translate into machine language; and finally, load and debug.

The first step, defining the task, is the most important step. To write a successful program you must understand the capabilities and limitations of the hardware and outline complete specifications of the task. A faulty problem design insures a faulty program.

Let's consider a simple problem. We have a microcomputer with a keyboard input and a CRT output. We want to use the keyboard and CRT as a four function calculator. This defines our task. The specifications are: perform addition, subtraction, multiplication and division. Use A + B = C format on the CRT display; entering only numbers. Calculation begins when the equal sign is entered. Now our specifications and limitations have been set. We're ready to flow chart our problem. This process often uncovers factors you may not have anticipated originally.

Once you clearly define the overall task, you reduce it to manageable blocks and flow chart it. In order to do this you'll need a thorough understanding of the problem, the hardware and the instruction set to be used. Fig 7 gives a flow chart for the problem described above. Once you've developed a flow chart you can become more specific, translating each block of the flow chart into mnemonics and finally into machine language. When you've reached this stage, you've written your program. It should resemble your microcomputer's documented software or software that you might see in microcomputer magazines.

Now to load your program into your computer and see it run – although it probably won't the first time (never has for me anyway). So you'll have to debug it. If you have a good troubleshooting mind, you might even enjoy examining your program and finding the difficulties. The problem will lie in one of two areas; errors in loading the program into memory or misuse of instructions.

To develop your programming skills, begin with simple tasks such as adding two numbers or moving a small block of data from one memory section to another. Then attempt more involved tasks. You'll be surprised at what you can learn from your mistakes as well as your successes. Look for magazine articles on your particular CPU to see what others have done and the techniques they have used. And by all means find another person involved in microcomputers – you can learn from each other.

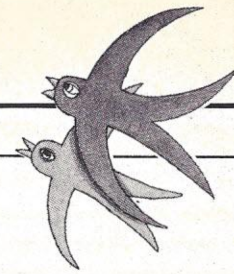
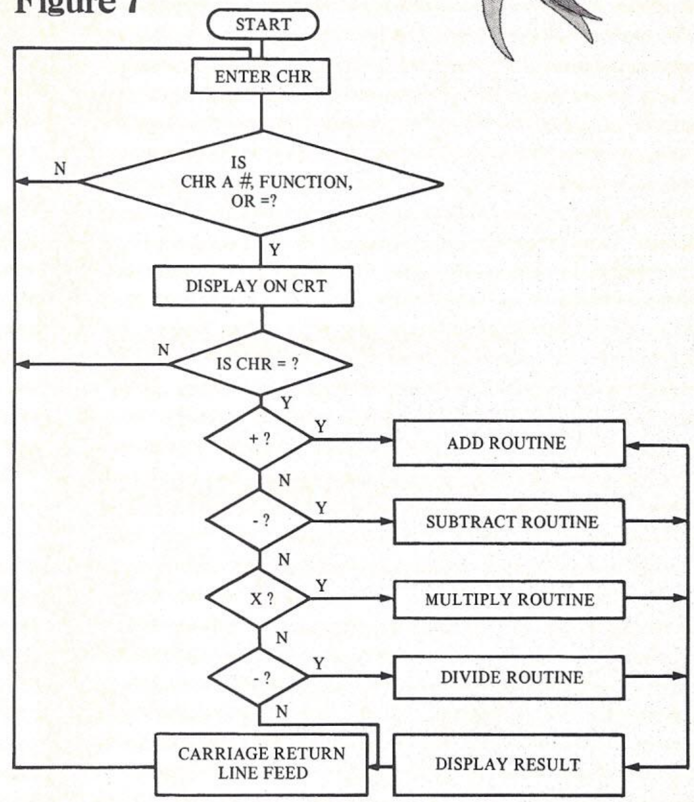


Figure 7



Test Yourself

This quiz is a learning aid. Some of the answers may be debatable but they'll force you to think through the various aspects of a computer. For this you'll be all the wiser.

1. A computer
   a. Handles one or two words at a time.
   b. Is composed of a CPU and I/O's only.
   c. Uses only 1's and 0's internally.
   d. Two of the above.
   e. None of the above.
2. Instructions
   a. Can be one, two or three words long.
   b. Have an address mode.
   c. Make up programs.
   d. Two of the above.
   e. Three of the above.
3. All memory
   a. Can be written into.
   b. Can be read from.
   c. Can forget.
   d. All of the above.

(Continued on p. 78)

BASIC SOFTWARE LIBRARY

Volumes VI & VII are finally released!

Volume VI

Yes and it still contains what was previously advertised. A fully disk interactive business package with A/R, Inv., A/P, ledgers, tax totals, payroll records, more.

As a bonus it also contains the Users Manual for our Firmware Ledger package. These 100 extra pages contain report formats, file creation routines and our very powerful program ACBS1 used to create the powerful file structured data base.

$49.95

Volume VII

Here is that Chess program you have been waiting for as well as a disk interactive Medical Billing package with patient history file.

Also included is our disk interactive Word Processing package (revision Ø).

$39.95

Add $1.50/Vol. for U.P.S. and handling except to APO and PO addresses. Foreign orders add $8/Vol. for air shipment — US dollars only. No purchase orders over $50.

VOLUME I —	$24.95
VOLUME II —	24.95
VOLUME III —	39.95
VOLUME IV —	9.95
VOLUME V —	9.95

OUR SOFTWARE IS COPYRIGHTED AND MAY NOT BE REPRODUCED OR SOLD.

Due to the numerous copyright violations on our earlier volumes — until further notice we are offering a REWARD leading to the arrest and conviction of anyone reproducing our software in ANY way without our written permission. This includes diskettes, paper and magnetic tape, cassettes, records, paper copies, etc.

SRI SCIENTIFIC RESEARCH

220P Knollwood
Key Biscayne, Florida 33149

Phone orders 800-327-6543
305-361-1153

AVAILABLE AT MOST COMPUTER STORES

BANKAMERICARD master charge

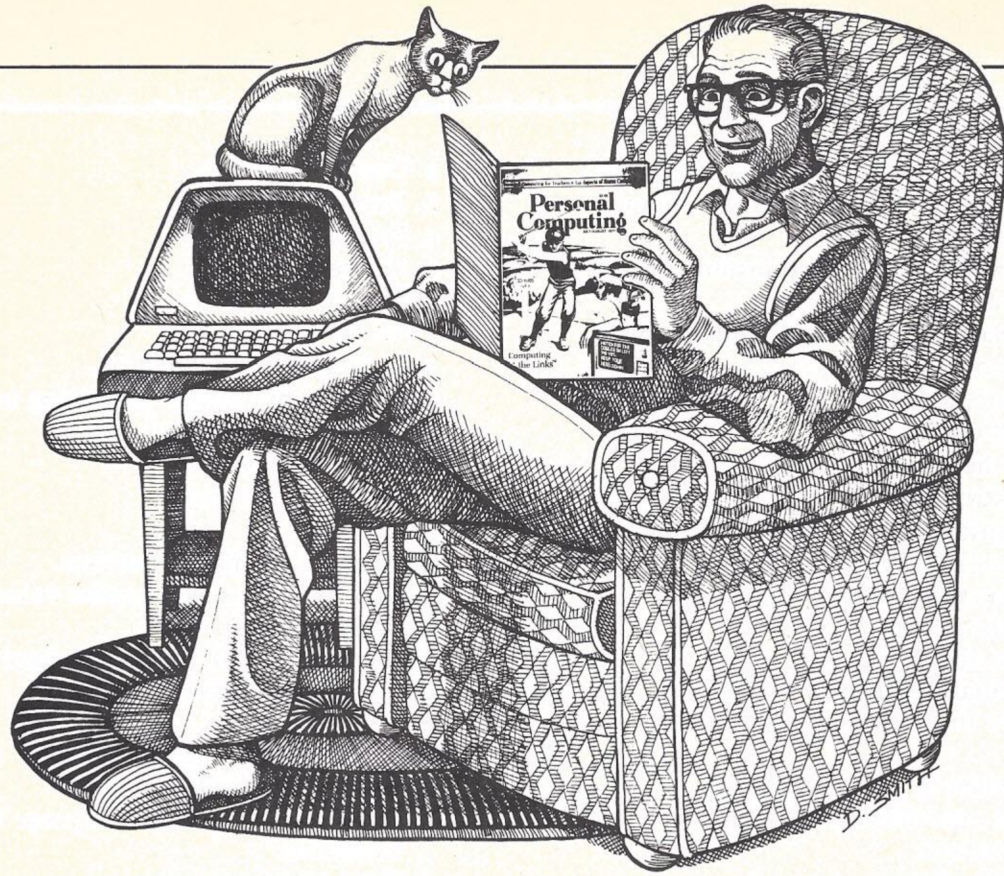
honored

(Continued from p. 76)

4. The memory is instructed to write
   a. By the Data Input port.
   b. By the Instruction Register.
   c. By the CCR.
   d. None of the above.
5. The Input Port
   a. Is used to enter a program.
   b. Looks like memory to the CPU.
   c. Is neither an airport nor seaport.
   d. Two of the above.
   e. All of the above.
6. The CPU
   a. Contains a program counter.
   b. Performs logic operations.
   c. Is stepped along by a clock.
   d. Two of the above.
   e. All of the above.
7. A Bus
   a. Is bi-directional, twice weekly and usually late.
   b. Parallel wires going to the same place.
   c. Is always eight conductors.
   d. None of the above.
8. Flags
   a. Are the results of ALU action.
   b. Are recorded as one bit of data.
   c. Are used by branch instructions.
   d. Two of the above.
   e. All of the above.
9. In the hexadecimal number system
   a. Sixteen binary digits are needed to express one hex digit.
   b. Two hex digits are required to express 16_{10}.
   c. The highest four digit binary number is equal to F_{16}.
   d. Two of the above.
   e. All of the above.
10. Software is
   a. Made up of instructions.
   b. Stored in ROMs, RAMs or PROMs.
   c. More useful if documented.
   d. Two of the above.
   e. All of the above.
11. The ALU
   a. Action effects the condition code register.
   b. Performs logic and arithmetic operations.
   c. Is controlled by the control register.
   d. Two of the above.
   e. Three of the above.

12. Instructions can be divided into five groups
   a. Control, data transfer, immediate, branch and logical.
   b. Logical, arithmetic, data transfer, control and branch.
   c. Branch, data transfer, math, pneumonic and control.
   d. Arithmetic, compare, control, complement and branch.
   e. None of the above.
13. Branch instructions
   a. Are conditional or unconditional.
   b. May be called jump instructions.
   c. May be one, two or three bytes.
   d. Two of the above.
   e. All of the above.
14. The immediate addressing mode
   a. Is used by the control group of instructions.
   b. Is used when in a hurry.
   c. Finds the operands in the second byte of the instruction.
   d. Two of the above.
   e. All of the above.
15. Return or no-return is an aspect of
   a. Control registers.
   b. Branch instruction.
   c. Program counters.
   d. Address modes.
   e. None of the above.
16. The operand
   a. Is data to be operated on.
   b. Is found by various addressing modes.
   c. Is sometimes implied.
   d. All of the above.
   e. None of the above.
17. Data
   a. Occupies memory locations.
   b. May have the same number value as some operator code.
   c. May be the first word of an instruction.
   d. Two of the above.
   e. All of the above.
18. Assembly language is
   a. Written in hexadecimal.
   b. Converted to machine language by a computer.
   c. Not a high level language.
   d. Two of the above.
   e. All of the above.

(Answers on p. 86)


Personal Computing
D. SMITH

Touching Down On: Computer Football

BY JOE ROEHRIG

Ever since graduate school I've been interested in computers and gaming. The curriculum at Iona Graduate School included courses in computer programming. Propelled into action by the inspiring courses, I dashed out and bought an IMSAI 8080 kit. After overcoming initial assembly problems (which is a story in itself) I eventually got the computer working. Armed finally with a system performing better than I had hoped for, considering all the trouble I had putting it together, I eagerly started the game programs. My interest in sports led me to develop basketball, horseracing,

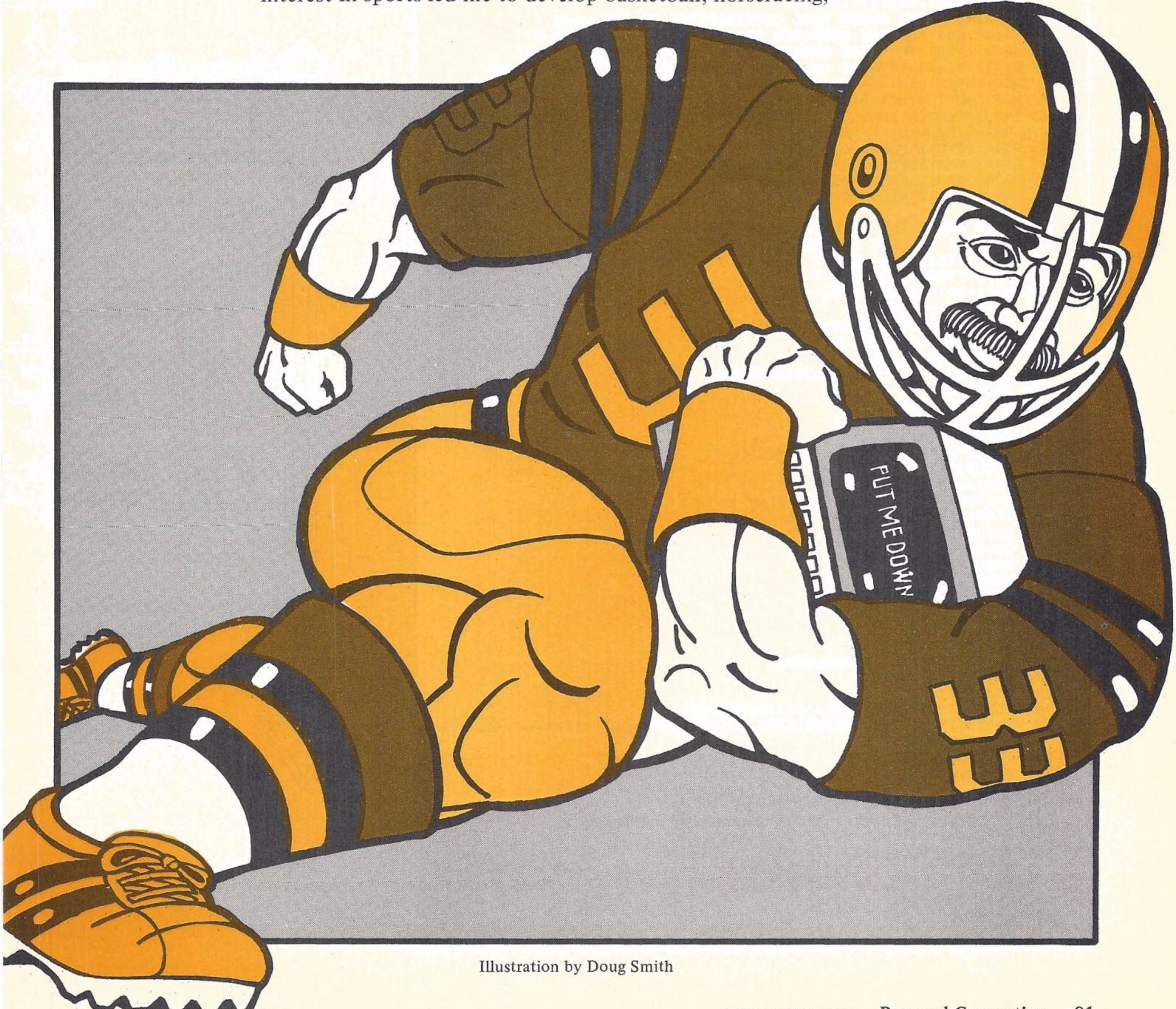


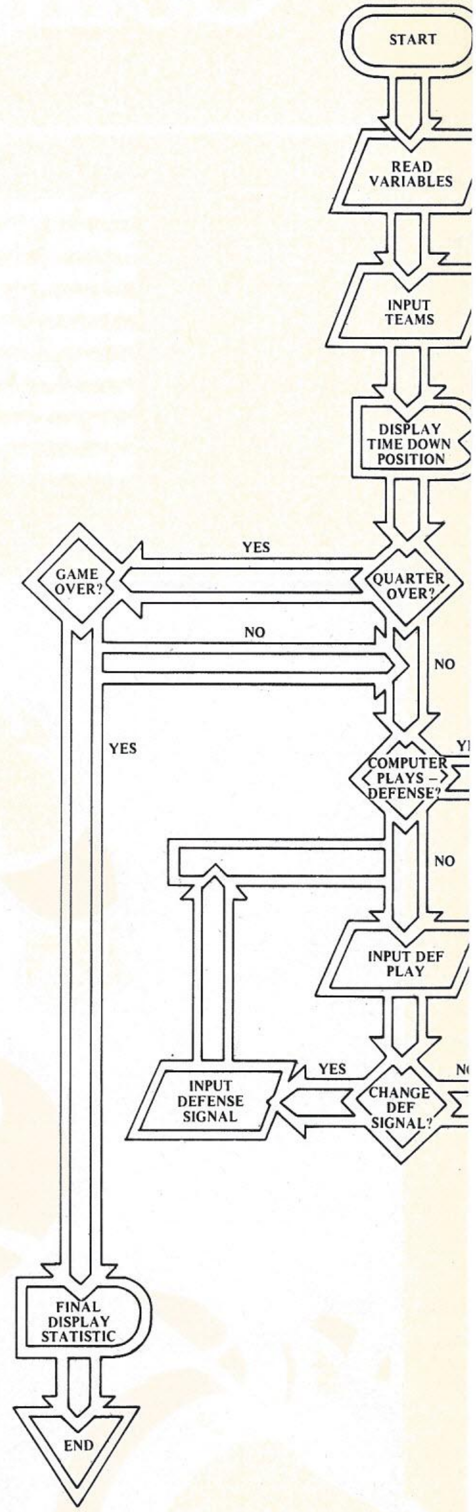
Illustration by Doug Smith

boxing and football games. Football captured my primary interest and that is the one I concentrated on.

The following program shows how the football game is played. This program does not utilize the data file described; therefore, a full 12K is available. BASIC, without trigonometric functions, takes 5277 words and the game uses 6370 words of memory. This leaves another 1093 words for additional refinements. The game can be played by two players or one (solitaire). If the solitaire method is selected, the computer plays against you. Data displayed includes time, position of ball, down and yards to go. The game is designed for a video terminal, therefore the data printed out was not limited. (You can also pit the computer against itself to test the program. A game played in this manner takes about 90 seconds to complete.) The input for the game is:

1. The computer asks for a 1 if you are playing solitaire or any other number for two players and two team names. (A 99 is input to play the computer.)

2. Next, the defense is asked for:
   0 = Standard defense.
   1 = Key against run.
   2 = Key on halfback run.
   3 = Key on fullback run.
   4 = Short yardage defense.
   5 = Pass defense.
   6 = Blitz.



Listing

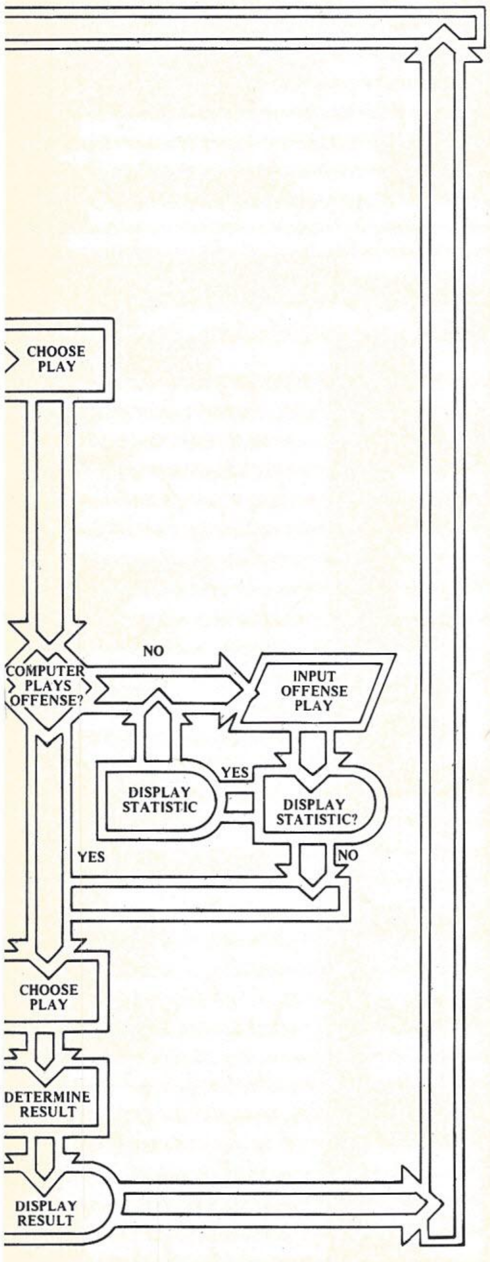
```
11 DIMP(11,7),M(10,10),R(13,10),Q(1,2),B(1,3,1),S(1),H(7)
13 DIMW$(163),W(1,17),N$(20)
15 W$(1,40)="END RUNHB GOES OUTSIDEPITCH TO THE HBHB "
16 W$(41,80)="UP THE MIDDLEFB DRAWFB UP THE MIDDLEFB P"
17 W$(81,120)="LUNGEQB SNEAKQB ROLLS OUT-RUNSSCREEN PAS"
18 W$(121,163)="SSHORT PASSLONG PASSPUNTFIELD GOALHBFBQBEND"
21 DEF FNT(Z)=INT(6*RND(0))+INT(6*RND(0))
31 DEF FNU(Z)=INT(10+(RND(0)*20))
35 INPUT"RANDOM NUMBER ? ",A\FORR1=0TOA\R2=RND(0)\NEXT
41 FORC=0TO7\H(C)=C\NEXT\A9=1
51 FORA1=0TO11\FORA2=0TO7\READP(A1,A2)\NEXT\NEXT
53 FORA1=0TO13\FORA2=0TO10\READR(A1,A2)\NEXT\NEXT
55 FORA1=0TO10\FORA2=0TO10\READM(A1,A2)\NEXT\NEXT
57 FORA1=0TO17\READW(0,A1),W(1,A1)\NEXT
100 N$="                    "
103 !"99 MAKES THE COMPUTER PLAY THE COMPUTER OR"
105 INPUT"1 TO PLAY THE COMPUTER OR 2 FOR 2 HUMAN PLAYERS ? ",C
111 INPUT"TEAM 1'S NAME ? ",A$
113 INPUT"TEAM 2'S NAME ? ",B$
115 N$(1,10)=A$\N$(11,20)=B$
121 IFRND(0)>.5THEN131
125 A1=0\A2=1\E8=0\E9=0\GOTO141
131 A1=1\A2=0\E8=1\E9=0
141 !"QUARTER #",A9
143 A3=FNU(Z)\T1=1\T4=T4+30\GOSUB881
145 GOSUB811
147 IFT4>=900THEN5111
151 IFA2=1ANDC=1THEN171
155 IFC=99THEN171
161 INPUT"DEF PLAY ? ",A4\GOTO182
171 GOSUB4111
182 IFA2=0ANDC=1THEN201
191 IFC=99THEN201\INPUT"OFF PLAY ? ",A5
193 IFA5<>14THEN211
195 GOSUB3111
197 GOTO191
201 GOSUB2111
211 IFA5<0THEN821
221 IFA5>13THEN821
231 IFA4<0THEN821
241 IFA4>8THEN821
251 IFA4=8THEN831
255 T1=T1+1\T4=T4+30\GOSUB881
256 IFA5>11THEN261
257 GOSUB891
261 GOSUB9000\!W$(R1,R2),\IF1>A5THEN281
271 ONA5GOTO291,291,291,311,311,311,321,321,411,411,411,1021,721
281 A7=3\GOTO331
291 A7=0\GOTO331
311 A7=1\GOTO331
321 A7=2
331 IFA6<>-99THEN341\R1=(A2+1)*10
333 !" FUMBLE ",N$(R1-9,R1)," RECOVERED "\GOTO621
341 GOSUB901
351 IFE1=0THENGOSUB981
361 B(A1,A7,0)=B(A1,A7,0)+1\B(A1,A7,1)=B(A1,A7,1)+A6
371 IFE1<=15THEN381\A3=0\GOTO1021
381 IFE1>5THEN641
391 GOTO500
```

7 = Prevent.

8 = Change defensive signals.

In the solitaire game, 8 is not needed. However, with two players and the de-

Football flowchart



(listing continued)

```
411 IFA6=-99THEN431
421 IFA6>=0THEN431\A7=2\!" QB FORCED TO RUN-NO PASS ",\GOTO341
431 Q(A1,0)=Q(A1,0)+1\IFA6<>-99THEN461
451 !" INTERCEPTED "\GOTO621
461 IFA6>0THEN481
471 T4=T4-15\GOTO500
481 GOSUB901
491 Q(A1,1)=Q(A1,1)+1\Q(A1,2)=Q(A1,2)+A6
493 IFE1>5THEN641
500 A3=A3+A6\IFA8>A3THEN531
505 !" PLAY GOOD FOR ",A6\GOSUB811
510 T1=1\E3=10\GOTO147
531 E3=A8-A3
533 !" PLAY GOES FOR",A6,"    ",T1," AND",E3
541 IF5>T1THEN147
621 A4=A1\A1=A2\A2=A4\A3=100-A3\T1=1\GOSUB811
631 GOTO147
641 T1=1\A4=A1\A1=A2\A2=A4\GOTO143
721 A4=INT(RND(0)*60)\A3=A3+A4
722 IFA3>100THEN741
731 !" MISSED "\A3=80\GOTO621
741 !" GOOD "\S(A1)=S(A1)+3\GOSUB961
751 GOTO641
811 !""\R1=((A1+1)*10)\!N$(R1-9,R1)," HAVE THE BALL ON THE",A3,
812 !" FIRST DOWN"
813 !TAB(5),"TIME GONE",T2,T3," PEROID",A9\!""
815 A8=A3+10\RETURN
821 !"INPUT ERROR"\GOTO151
831 FORA5=0TO7\GOTO841
840 !"BAD SIGNAL, RE-INPUT ",
841 !"DEF SIG",A5\INPUT"  ? ",H(A5)
851 IFH(A5)<0THEN840
861 IFH(A5)>7THEN840
871 NEXT\GOTO161
881 T2=INT(T4/60)\T3=T4-(60*T2)\RETURN
891 A4=H(A4)\A4=P(A5,A4)\A6=FNT(Z)\A6=R(A4,A6)\RETURN
901 E1=A3+A6\IFE1<100THEN905\A6=100-A3\!" TOUCHDOWN"\GOTO911
905 E1=0\RETURN
911 S(A1)=S(A1)+6\IF.9<RND(0)THEN931
921 S(A1)=S(A1)+1\!" POINT AFTER GOOD"\GOTO941
931 !" POINT AFTER BLOCKED"
941 GOSUB961
951 E1=9\RETURN
961 !"SCORE: ",N$(1,10),S(0),"    ",N$(11,20),S(1)
971 !"*****"\RETURN
981 E1=A3+A6\IF0<E1THEN991\A6=0-A3\!" TOUCHBACK"\GOTO1001
991 E1=0\RETURN
1001 S(A2)=S(A2)+2\E1=19\GOSUB961
1011 RETURN
1021 E1=30+INT(RND(0)*40)+A3\IFA3>100THENA3=80
1031 GOTO621

2111 E4=FNT(Z)\IFT1<>1THEN2113\E3=8\GOTO2171
2113 E3=A8-A3
2121 IFT1<>4THEN2161
2131 IF3<=E3THEN2141\E3=10\GOTO2171
2141 IF60<=A3THEN2151\A5=12\RETURN
2151 A5=13\RETURN
2161 E3=9
2171 A5=M(E3,E4)\RETURN
3111 !""\FORN=0TO1\R1=((N+1)*10)\!N$(R1-9,R1),S(N)\!"::::::::::"
3113 !"PLAYER RUSHES YARDS AVERAGE"
3121 FORN1=0TO3\IFB(N,N1,0)=0THEN3127
3124 E4=N1+14\E3=B(N,N1,1)/B(N,N1,0)
3125 R1=W(0,E4)\R2=W(1,E4)\!W$(R1,R2),TAB(7),
3126 !%5I,B(N,N1,0),%6I,B(N,N1,1),%8F3,E3
3127 NEXTN1\!""
3132 !"PASSES COMPLETIONS YARDS %COMPLETED AVE.GAIN"
3133 IFQ(N,0)=0THEN3155
3141 E3=Q(N,1)/Q(N,0)\E4=Q(N,2)/Q(N,0)
3151 !%6I,Q(N,0),%12I,Q(N,1),%6I,Q(N,2),%11F3,E3,%9F3,E4
3155 !""\NEXTN\RETURN
4111 IFT1<>1THEN4121\E3=0\GOTO4321
4121 IFT1<>4THEN4131\E3=7\GOTO4321
4131 E3=0\IFT1=3THENE3=3
4132 E4=A8-A3\IFE4<=9THEN4134\E5=1\GOTO4311
4134 IFE4<=5THEN4136\E5=2\GOTO4311
4136 E5=3
4311 E3=E3+E5
4321 A4=FNT(Z)\A4=M(E3,A4)\RETURN
5111 T4=0\IFA9<>2THEN5125\A9=3\A1=E9\E2=E8\GOTO141
5125 IFA9=4THEN5141
5131 A9=A9+1\!"QUARTER #",A9\GOTO147
5141 GOSUB961
5151 GOSUB3111
```

(listing continued)

```
5711 DATA0,1,2,2,0,2,6,3
5712 DATA2,1,5,3,4,3,3,3
5713 DATA2,1,6,3,4,3,4,3
5714 DATA2,1,6,1,6,2,4,3
5715 DATA5,2,3,1,2,3,0,4
5716 DATA2,1,1,6,6,2,4,3
5717 DATA2,2,3,5,6,2,3,4
5718 DATA1,5,5,5,6,2,3,4
5719 DATA2,1,2,2,3,1,2,4
5720 DATA7,7,8,8,7,8,7,7
5721 DATA8,9,9,9,9,10,11,7
5722 DATA10,13,13,13,13,12,11,7
5730 DATA99,13,11,9,7,5,3,1,-1,-3,-10
5731 DATA-99,-99,6,5,4,3,2,1,0,-2,-3
5732 DATA-8,-4,0,2,3,4,5,6,8,12,16
5733 DATA-4,-3,-2,3,4,5,6,7,9,11,17
5734 DATA-6,-4,-2,0,4,6,8,10,14,18,25
5735 DATA-4,-3,-2,0,1,2,2,3,4,5,6
5736 DATA5,4,3,2,1,0,-1,-2,-3,-4,-5
5737 DATA52,0,0,9,6,5,4,0,3,2,1
5738 DATA-5,-99,0,9,6,5,4,0,2,-5,-5
5739 DATA-5,8,18,0,0,12,11,0,0,32,-5
5740 DATA-10,-5,18,0,0,12,0,-99,-8,32,51
5741 DATA-15,-10,-5,50,-99,0,0,18,-5,-10,-5
5742 DATA100,50,-15,-99,0,0,0,28,18,-5,-10
5743 DATA-10,50,-99,0,0,0,28,18,100,-5,-10
5750 DATA0,0,5,1,1,6,5,1,5,0,0
5751 DATA6,6,5,5,1,6,1,5,5,6,6
5752 DATA2,3,3,2,1,6,1,3,2,5,5
5753 DATA5,6,1,1,0,6,0,1,1,5,6
5754 DATA6,6,7,7,7,6,7,7,7,6,6
5755 DATA2,3,3,2,1,6,1,3,2,5,6
5756 DATA5,6,1,1,0,6,0,1,1,5,5
5757 DATA6,2,3,1,4,6,4,1,2,3,6
5758 DATA0,9,10,1,4,11,2,9,10,1,0
5759 DATA0,1,9,10,11,4,2,1,9,10,0
5760 DATA11,11,8,3,5,6,3,5,8,11,11
5771 DATA1,7,8,22,23,37,38,53,54,60,61,76,77,85,86,93
5772 DATA94,110,111,121,122,131,132,140,141,144,145,154,155,156
5773 DATA157,158,159,160,161,163
9000 R1=W(0,A5)\R2=W(1,A5)\RETURN
```

Run

```
RANDOM NUMBER ? 22
99 MAKES THE COMPUTER PLAY THE COMPUTER OR
1 TO PLAY THE COMPUTER OR 2 FOR 2 HUMAN PLAYERS ? 1
TEAM 1'S NAME ? NY JETS
TEAM 2'S NAME ? LA RAMS
QUARTER # 1

LA RAMS     HAVE THE BALL ON THE 26 FIRST DOWN
     TIME GONE 0 30 PEROID 1

DEF PLAY ? 8
DEF SIG 0
  ? 7
DEF SIG 1
  ? 6
DEF SIG 2
  ? 5
DEF SIG 3
  ? 4
DEF SIG 4
  ? 33
DEF SIG 4
  ? 2
DEF SIG 5
  ? 1
DEF SIG 6
  ? 0
DEF SIG 7
  ? 3
DEF PLAY ? 3
LONG PASS PLAY GOOD FOR -10      2 AND 20
DEF PLAY ? 4
FB DRAW PLAY GOOD FOR 5     3 AND 15
DEF PLAY ? 4
FB DRAW PLAY GOOD FOR 4     4 AND 11
DEF PLAY ? 7
PUNT
NY JETS     HAVE THE BALL ON THE 75 FIRST DOWN
     TIME GONE 2 30 PEROID 1

OFF PLAY ?
```

fense being asked for first, 8 is used to disguise the defense. Each time a player goes on defense, the offensive player should not look and the defensive player can put in the number 8. The computer will then solicit numbers for defenses 0 to 7. Therefore, all defenses can be disguised (zero can be two, for example.)

3. Next, the offensive play is selected:
   - 0 = End run.
   - 1 = Halfback on an outside run.
   - 2 = Halfback on a pitch out.
   - 3 = Halfback up the middle.
   - 4 = Fullback draw.
   - 5 = Fullback up the middle.
   - 6 = Fullback short yardage situation.
   - 7 = QB sneak.
   - 8 = QB roll and run.
   - 9 = Screen pass.
   - 10 = Short pass.
   - 11 = Long pass.
   - 12 = Punt.
   - 13 = Field Goal.
   - 14 = Gives offensive statistic and then asks for another play selection.

NOTE: None of the above instructions are written out by the computer in order to save memory space.

For fast edits to the program:

LINES	PURPOSE
11-161	Set-up of data, kickoff function and random numbers function.
171-257	Play selections and validation of play selected.
261-641	Actual play calculation.
721-751	Field goal.
811-815	Prints information.
821	Error message.
831-871	Change defense signals.
881	Calculate time.
891	Play yardage calculation.
901-951	Touchdown check and extra point try.
961-971	Prints score.
981-1011	Touchback check.
1021-1031	Punt.
2111-2171	Computer offensive play selection.
3111-3155	Prints statistics.
4111-4321	Computer defensive play selection.
5111-5161	Changes quarter.
5711-5722	Table to compute result of defensive and offensive play selection.
5730-5743	Table for yardage gained.
5750-5760	Computer play selection.
5771-5773	Size of print messages.
9000	Calculates size of message to be printed.

Scaling the Cliffs to Computer Mastery: Quiz Answers

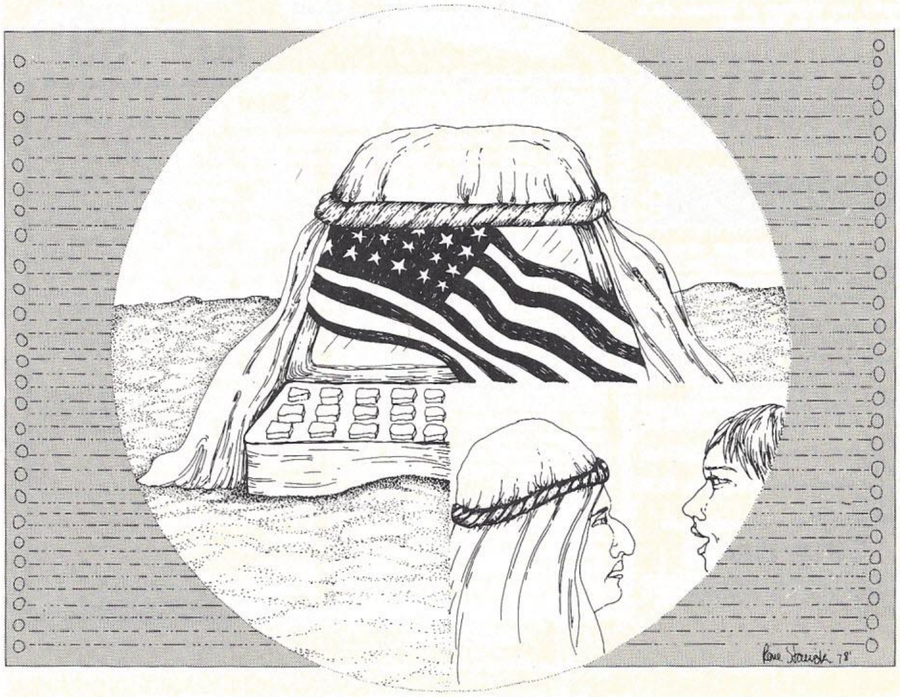
From pp. 76, 78 & 79.

1. c. A computer handles only one word at a time, though a word may contain more than one byte, depending on the particular CPU. A computer has memory in addition to a CPU and I/Os.
2. e.
3. b. The CPU can only write into RAMs. Under special conditions the CPU can write into PROMs. ROMs cannot be written into by a CPU, they are manufactured with their content fixed. RAMs forget when power is removed but ROMs and PROMs retain their contents when power is turned off.
4. d. Memory is instructed to write by the CPU via the control bus.
5. e. Input ports are used to enter programs, but they're also used to input data from other devices you may connect to your computer.
6. e.
7. b. A bus may be any number of conductors. The data bus in an eight bit word CPU has eight conductors. The address bus in a

64K memory computer has 16 conductors. The number of conductors in the control bus varies with each CPU.
8. e. B may have caused you trouble. Each flag represents only one bit in the Condition Code Register.
9. d, b and c are correct. Sixteen binary digits will express two hex digits.
10. e.
11. d, a and b are correct. The instruction register controls the ALU.
12. b.
13. e.
14. d, a and c are the ones.
15. b.
16. d.
17. d, a and b are correct. The first word of an instruction is not data, it is an opcode.
18. d, b and c are correct. Assembly language is written in mnemonics. Machine language uses either binary, octal or hex.

Computers in Developing Nations: A Cautionary Tale

A mistake easily made is to look at the introduction of computing in developing nations from the perspective gained by the experience of introducing computers in well developed industrial countries. Such mistakes are easily made because meetings of "experts" from developed and undeveloped countries often turn out to be meetings of alumni from the same group of universities.



In order to learn the nitty gritty of computing in developing countries, computer experts ought to see firsthand what problems exist right down to the level of coders and end-users. The journey is rewarding for those computer scientists who have an anthropological bent. It will make no sense to anyone else. (The major obstacles to rational use of technology are more often social then mechanical.) While my tale can hardly be compared to a professional study of other cultures from which we gain understanding of our own, there is more to the tale than meets the eye.

The story begins in 1972 when, in response to an invitation by the then Minister of Health and Welfare of the Shiekdom of Kuwait, I cast an advising eye upon some of Kuwait's newly installed computing ventures. It did not take very long to spot the major obstacles to a satisfactory flow of work which quite obviously was not materializing. Kuwait had gone the 360 way. The best and most modern of IBM products were purchased. Among the benefits was PL/I. Now PL/I resembles more colloquial English than any other computer language, and deviation from the compiler's proper grammar results in streams of messages returning diagnostics to the programmer. This is a felicitous situation for programmers who use English with ease, but not exactly the best circumstances for sons of nomads, bedouins, and fellahin whose knowledge of English is neither deep nor fluent. To the person who does not speak English well, the writing and debugging of programs in PL/I may be formidable. The task was not made easier by the quasi-nationalized management problems for which the computers are needed in Kuwait, such as to prepare utility-service bills for all inhabitants or to manage pharmaceutical inventories for an integrated set of clinics and hospitals comprising a comprehensive national health service (a health service that, incidentally, would be the envy of any Western country). These jobs require handling of large files with complex data structures and demand a fairly high level of competence in PL/I. Whether or not this competence existed among the programmers was not quite clear. Certainly competence in English comprehension was lacking.

The language problem obviously was not limited to Kuwait. I made a cursory check with IBM's Middle East headquarters in Beirut (before the tragic political events overtook that city, Beirut was the major center for what was called the "English-speaking Arab world", by which I took to mean the Arab world that spoke English but not exactly fluently). This revealed that IBM dominated the Arab market and that for all practical purposes, PL/I was the standard programming language from Gibraltar to Istanbul. I had a suspicion (well verified shortly thereafter) that similar programming problems hounded most installations from the legendary pillars of Hercules to the fabled Bosphorus.

But the first part of the problem, that of language comprehension, obviously was amenable to a solution. There appeared to be few reasons why the diagnostic messages could not be translated into Arabic and produced on the Arabic print chains (which were available in all installations). As for language statements, a preprocessor or interpreter could enable the PL/I compiler to deal with a program in which all statements were acceptable in Arabic or in English. Certainly given the limitless wealth of oil, there was no reason

why the Arab world could not be supplied with an Arabic version of PL/I. It was even fortunate that an Arab student was among my graduates at that time, completing his thesis on a software problem for which PL/I was the major language. My report to the sponsor and a recommendation to IBM International was to furnish the sensible and needed extension of software by providing a translation for all diagnostic messages into Arabic and adding a preprocessor to the PL/I compiler that would accept source statements in Arabic. What could be easier?

What followed was far from simple. It was a curious sequence of events which showed the complexity of the human psyche to its best advantage. Immediate and bitter protest to the proposal came from a source which I least expected would object

Computer experts ought to see firsthand the problems of coders and end-users.

—the programmers themselves. Technically competent individuals are a scarce resource in developing countries, and those who in addition must speak English to perform a job make up a very small subset. Thus the programmer's ability to speak English was an important factor assuring job security and a high level of pay. Facility management was no less opposed to the proposal although less outspoken against it. Again, while the advantages of an Arabic compiler could not be denied, there were greater advantages requiring management to be conversant in English. Politically the usefulness of the language restriction was quite apparent as all top level management positions had to be held by bedouins, natives of Kuwait, few of whom spoke English. The effective management of facilities was conducted by immigrants from Egypt or Palestine whose position to a large extent depended upon

their ability to supervise English-speaking employees in the task that required the mastery of English. The Arabic compiler would have eliminated the leverage exercised by the associate manager over a variety of management-level positions.

Nevertheless, IBM International agreed that there was a great deal of merit to the idea of supplying some sort of translation mechanism such as a tree-structured dictionary or encyclopedia for the purpose of guiding the perplexed Arab programmer through the intricacies of semantic pitfalls which systems-oriented language offered. While clearly not an optimal solution, producing the reference dictionary would have been of some use to the region. But then another obstacle arose quite unexpectedly from the so-called French-speaking Arab world.

The French-speaking Arab world by extension is the area of North Africa in which French is spoken and with a somewhat greater fluency than English is spoken in the Middle East. It turned out however that the French-speaking Arab world also used IBM equipment and PL/I. Within the relatively large supply of French-speaking technically trained Arabs, those who speak English occupy again a more favorable position and their objections to any tampering with the status quo was indeed fast and vehement. The strenuous resistance by the Paris office of International IBM scuttled the project altogether. And there the battle rests.

The inanity of this situation is by no means limited to the Arabs. In 1972 while attending to a similar need of the Israelis, I suggested that although Israeli programmers were more fluent in English than their Arab counterparts, they too could have profited by a Hebrew preprocessor and translation of diagnostic messages to their mother tongue. Again, the objection was no less vehement among the Israelis than it had been among their Arab counterparts. In fact, the wording of arguments against the proposal reflected similar feelings among Arabs and Is-

raelis. (So striking was the similarity of arguments that for a mad moment I suspected a secret cabal between Israeli and Arab programmers—and for all I know there might be one.)

Since those days I have run into the same or similar attitudes on other occasions. The last time was during a meeting sponsored by the International Atomic Energy Agency which brought together radiation therapists and computer experts from South American and Asian countries. Radiation therapy has profited immeasurably from the computer-produced radiation-treatment plans and there are a number of systems which will permit near-optimal plans to be devised in interactive procedures. Using any of these systems unfortunately depends upon a thorough knowledge of English. Again the problem in developing countries appears to be the shortage of technically trained English-speaking personnel who may be spared from other computing tasks to work with the radiation therapists on treatment planning problems. My suggestion of providing interpreters for the various packages that would enable them to be manipulated in the native tongue of the technicans turned out to be as welcome as a case of the yaws. While hardware systems especially put together to permit radiation treatment planning are now being installed the world over, accompanying software consistently uses English, leading to an increasing competition for the already short supply of English-speaking technicians the world over. This may all be to the good of the English-speaking part of the globe but what benefits accrue from this to the rest of mankind?

I was very surprised that the discussion summarized by E. Robertson [1] of the panel on "Computers in Developing Nations" does not even touch upon the crucial problem of translating software into NCLs (Native Compatible Languages). This is not to say that software in "native"

"The problem in developing countries is the shortage of technically trained English-speaking personnel..."

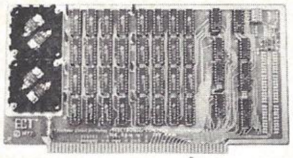
languages will open new channels for better planning and allocation of scarce resources among Third World countries. But it will help upgrade a large number of their citizens and gives them access to an important new technology.

There is yet one more implication of this same language issue which relates to us, here, in our highly developed technical society. Are we not also to some extent victimized by the same forces that prevent the implementation of more useful compilers in underdeveloped nations? Fortran and Cobol have maintained an iron grip on programming in Canada, the United Kingdom, and the United States, even though, for all practical purposes, PL/I is a much easier language to learn, to use, and in which to develop software. We find Fortran taught to half the students in business schools on the North American continent and the teaching of Fortran in general, with the almost natural way of teaching bad programming habits, is deeply ingrained in many otherwise foresighted and topnotch universities. Why did naturally modular PL/I not take hold more quickly? Why is the use of PL/I resisted by programmers and management alike? Also why do programmers, who "speak" PL/I, again resist the introduction of specialized or general languages more suitable to their tasks? To what extent are the motivations similar for faculty in a business school or in a chemistry department who will fiercely defend a Fortran territory and require their students to learn a language which is poorly suited for character and string manipulations?

However, this cautionary tale is about developing (or what we used to call underdeveloped) countries, and surely(?) no lessons from these would apply to us here. **PC**

1. Computers in developing nations. *Computers and Society* 7 (1976), 7-9.

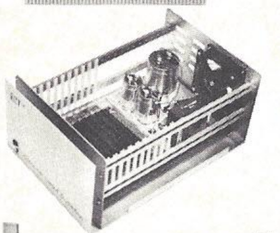
Reprinted from **Communications of the ACM,** *Theodor D. Sterling, Simon Fraser University, Burnaby, B.C., Canada. Volume 20, No. 12. December, 1977.*



16K RAM

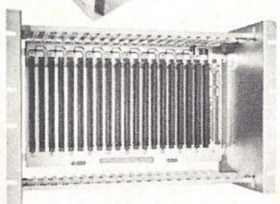
FULLY STATIC

$450 INTRODUCTORY KIT PRICE



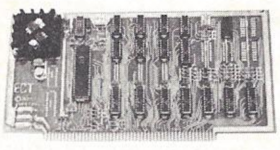
10 SLOT TABLE TOP MICROCOMPUTERS
TT-8080 KIT $475

SYSTEM W/16K & I/O
TT-8080-S KIT $1125



CARD CAGE & MOTHER BOARD
ECT-100 KIT $100

WITH CONNECTORS & GUIDES
ECT-100-F KIT $200



CPU'S, MEMORY
MOTHER BOARDS
PROTOTYPING BOARDS
EXTENDER CARDS
POWER SUPPLIES

DEALER INQUIRIES INVITED *SHIPPING EXTRA*

ELECTRONIC CONTROL TECHNOLOGY

P.O. BOX 6, UNION, NEW JERSEY 07083
(201) 686-8080

CIRCLE 32

IT'S A GREAT BIG COMPUTER WORLD
But You Only Need

THE COMPUTER CORNER

- **SOL** – A New Dawn Is Here!
- **IMSAI** 8080
- **POLY** – 88
- **TDL** Z-80
- Memories & I/O Boards

- Computer Book Service
- Magnetic Tapes & Disks
- Full Line of Magazines
- Brain Games & Puzzles
- Workshops & Club Information

Visit **THE COMPUTER CORNER** for all your computer needs. Stop in and browse – you'll like our personal service.

THE COMPUTER CORNER
White Plains Mall – Upper Level
200 Hamilton Avenue
White Plains, New York 10601

Tel: (914) WHY - DATA

Ample Parking
10-6 Daily & Saturday
10-9 Thursday

CIRCLE 34

INQUIRE TODAY!!

We Ship From Stock

Business Systems Are Here Now!

COST EFFECTIVE HARDWARE • PERIPHERALS • SOFTWARE

Call our Mr. Amunson

Informative assistance in defining your objectives and choices.

For Engineers and Serious Hobbyists

BREADBOARDING • WIRE WRAP • DESIGN EQUIPMENT • TEST EQUIPMENT • COMPONENT PARTS • POWER SUPPLIES

• AP Products • OK Machine • Iso-Tip • B&K PRECISION • CONTINENTAL SPECIALTIES • NON-LINEAR SYSTEMS • PANVISE • VECTOR

Our 25th Year

ATLAS ELECTRONICS CORP

1570 3rd AVE. (88th St.) NEW YORK, N.Y. 10028
212-427-4040

Send $1 for complete set of catalogs (refunded on 1st purchase)

**We ship nationwide & overseas*

CIRCLE 33

TWO BYTES ARE BETTER THAN ONE

COMPARE 16-BIT COMPUTERS

THE
TECHNICO
SUPER STARTER SYSTEM
AND
SUPER SYSTEM 16

V.S.

THE HEATH H-11

SAVE OVER $1000
SEND FOR FREE BROCHURE

TECHNICO **1-800-638-2893**

9130 RED BRANCH RD. COLUMBIA, MD 21045
DEALER INQUIRIES INVITED

CIRCLE 35

PROGRAMMING HANG-UPS

Hands on

Programming a computer is like driving a car. To becom[e] an expert, you must put your hands on that mean machi[ne] and try it yourself. Every time you read a paragraph abo[ut] programming, try it: type the examples and look, look, s[ee] the computer run. Invent your own variations. Write you[r] own programs. Make it solve problems from the rest of yo[ur] life. The computer is a fantastic toy – play with it.

Hang around with your computer. Communicate with i[t] every day. At first, that will be harder than talking with a cat or tree, because the computer belongs to a different species, a different kingdom; but keep trying. Get to know it as you know your best friend.

Test

When you've written a program, *test* it: type RUN and see if it works. When the computer starts griping, you know the program doesn't work, and you can hunt for the bug.

If the computer does *not* gripe, your tendency will be to say "Whoopee!" Don't cheer too loudly. *The results the computer is printing may be wrong.* Even if the results look reasonable, don't assume they're right: the computer's errors can be subtle. Check its answers by calculating on a piece of paper. But even if its answers are correct, don't cheer. Maybe you were just lucky. Type different input, and see whether your program still works. Chances are, something you input will make your program go crazy or print a wrong answer. Your mission: find that input.

Debug with dignity

If you write your own program and run it, chances are it won't work. Your first reaction will be to blame the machine; don't! The probability is 99.99% that the fault is yours. Your program contains errors. A programming error is called a *bug.* Your next task is to *debug* the program, which means getting the bugs out.

Bugs are common; top-notch programmers make errors all the time. If you write a program that works perfectly on the first run and doesn't need debugging, it's called a *gold-star program,* and means you should have tried writing a harder one instead!

Beware of "wares"

Your computer center needs four ingredients: hardware, software, printware and brainware. *Hardware* means the computer machinery itself. *Software* means programs. *Printware* means printed explanations of how to use the computer. *Brainware* means brainy people (like yourself) who know how to handle the computer.

You need all four of the "wares" – and they all cost money. If someone offers to sell you a "complete computer system," *beware:* the price may include just the hardware. Ask how much software, printware and brainware are included.

Avoid half-hearted hardware

Some think a "computer" is just the central processing unit. But don't be fooled by a slick salesman or a jazzy ad. Check whether the price also includes the main memory, auxiliary memory, input/output devices, cooling devices, power regulators, off-line equipment and interfaces. And if the salesman says, "Yeah, sure," check into the *quality* and *quantity* of each.

Buying a microcomputer? Ask whether the price is for a "kit" or for a "computer assembled and tested." To be safe, buy the computer assembled and tested. Though a kit is cheaper, it's hard to put together correctly: either you won't follow the wiring diagram correctly, or the static electricity in your body will "blow up" the CPU, or you'll spill a drop of solder and create a short circuit, or the kit will contain a defective or missing part. Hobbyists estimate that for every ten computer kits that have been sold, nine still aren't working.

Simplify

Too often, programmers embark on huge projects they never finish. Once you have an idea of how hard your project is, *simplify it.* Instead of making the computer play a perfect game of chess, how about just programming it to play without cheating?

In other words, *pick a less ambitious, more realistic goal* which, if achieved, will make you feel good and will be a stepping stone to your ultimate goal. Finding a bug in a program is like finding a needle in a haystack: finding the needle is easier when the haystack is small.

Are numbers good or bad?

Since the computer handles numbers easily, it encourages you to reduce problems to numbers. That's both good and bad. Quantification is good because it forces you to be precise. It's bad because some people are starting to make it a goal in itself, forgetting that it is but a tool to other ends. Counting the words that Shakespeare wrote is of no value in itself: it must be put to some use. In both the humanities and the social sciences, the unfortunate motto of the future will be, "If you can't think, count." Cynics have remarked, "The problem with computers is that they make meaningless research possible." Since only quantifiable problems can be computerized, the danger arises that, in a burst of computer enthusiasm, people will decide that non-quantifiable problems aren't worth investigating.

People are being reduced to numbers: telephone numbers, social security numbers and zip codes. When you start treating another human as just a wrong telephone number and hang up in his face, something is wrong.

Escape our culture

To create above-average computer art, write a weird program whose consequences you don't fully understand, make the computer obey it, and look at the computer's drawing. If the drawing looks nice, keep it and call it "art" – regardless of whether the drawing was what you intended. Maybe it resulted from an error, but who cares, so long as it looks interesting? Anything interesting is art. If the drawing "has potential" but is somehow unsatisfactory, change a few lines – or run the program again unchanged and hope the random numbers will fall differently. The last thing to invent is the title. Whatever the drawing reminds you of can be the title.

That method may seem a long way from da Vinci, but it's how most computer art is created. The rationale? Don't over-plan; let the computer "do its own thing"; it will give you art that escapes the bonds of human culture and reveals the unknown.

"Big Brother"

Even if the computer doesn't store damaging information about you, the mere *thought* that your every action is being recorded hurts, because it makes you act more conservatively. You may fear adopting a good but unusual lifestyle, because anything "different" about you will look bad on the computerized records used by banks, credit-card companies, insurance companies, and other conservative institutions.

What's harmful is not that Big Brother is watching, but that you *feel* he's watching. You are subjugated.

Crime wins praise

Since computer crime requires intelligence, even if you get caught you'll be admired. The usual reaction is not "What a terrible thing you've done!", but rather, "Gee, you must be smart. How did you do it?"

A bright button-down computer criminal who steals $100,000 electronically will usually get a lighter sentence than a dude who has to resort to a gun to get $1000. Is that justice?

Will computers unemploy us?

When computers do our work, will enough work remain to keep us busy? Don't worry: when no work is necessary, we humans have an amazing talent for inventing it. That's the purpose of Madison Avenue – to create new longings. Instead of significantly shortening the work week, we've always opted for a work week of nearly equal length but devoted to more luxurious ends. That's the gung-ho Protestant work ethic we're so famous for.

In the long run, computers will change our work without reducing it. But for the next decade or two, as society shifts to computers, many of us will temporarily lose our jobs.

Reprinted from **The Secret Guide to Computers.** Four volume guide, $11.50. Russ Walter, 92 St. Botolph St., Boston, MA 02116.

CLASSIFIEDS

Personal Computing Classified Rates

Individual: $1 per word; address set free; first two words set in all capital letters for free. If you want **bold** type, add $1 for each word set bold. Add $5 if you want replies sent to us and forwarded to you.

Company: 1X: $60/inch; 3X: $55/inch; 6X: $50/inch. Display ads accepted only if camera-ready materials submitted. Agency commission paid on camera-ready ads and after the first insertion of a publisher-set ad if subsequent insertions run without changes. Publisher sets ads in format and attempts to fit copy into space requested.

Advertisement accepted, however, with understanding that space may be larger than requested. Minimum billing size one inch; ads billed in half-inch increments thereafter. Frequency rate applies to publisher-set ads only if they run without changes in subsequent issues.

To place an ad, send copy to Classified Ad Dept., **Personal Computing,** *1050 Commonwealth Ave., Boston, MA 02215*

ADVERTISERS' INDEX

SALES OFFICES

Northeast: George Palken, 1050 Commonwealth Ave., Boston, MA 02215; (617) 232-5470 • **Mid-Atlantic**: Arthur Daks, Benwill Publishing Corp., 299 Madison Ave., N.Y., N.Y. 10017; (212) 661-0360 • **Mid-West**: Ralph Petersen, 1 Wheaton Ctr. No. 1706, Wheaton, IL 60187; (312) 653-2480 • **Southeast**: Harold Short, Parker & Short, 1 Lakeside Office Park, Wakefield, MA 01880; (617) 246-2293 • **West**: Yuri Spiro & Carol Stagg, 823 Enchanted Way, Pacific Palisades, CA 90272; (213) 454-0624 • **Japan**: Hiro H. Irie, International Business Corp., 11-8, Narita-Higashi 1-chome, Suginami-KU, Tokyo 166; Phone: (03) 311-1746.

Verbatim™

The whole message is quality.

You want to record your message verbatim—word for word—whether it's bits, bytes or "Dear Folks" translated into word processor language.

Our objective in manufacturing recording media for the electronics industry—digital tape cassettes, floppy disks, mag cards, computer cartridges—is to give you the finest, the best, the most dependable, the most cost-effective.

That means rugged, long-lived, abrasion-resistant recording media with superior magnetic qualities. If we made tires, they'd be steel-belted radials.

We delivered our first digital grade certified tape cassettes back in the beginning, 1969. We made the first commercial 3740-compatible floppy disks that didn't bear IBM's name. And the first Flippy® reversible flexible disks with *anyone's* name on them. The first mini data cassette is ours. And we've got the newest miniature flexible disk, the MD 525.

Now, **Verbatim** media. It's a new formulation of ferric oxides, an advanced macromolecular binder system to adhere it to the tough polyester film, and a process control system that demands over 200 separate quality checks before the material is cut, packaged, and certified to be 100% error-free.

The final quality check? "Make it pretty!" Our production people tell us that magnetic recording media is one of the rare instances in manufacturing where aesthetic appearance translates directly into final product quality. It has to look beautiful to work beautifully.

We have the formulas, the machines, the technology to make high quality recording media. But it takes the best people in the industry to deliver Verbatim disks, cards, cartridges and cassettes. You'll find them at your favorite retail computer store.

Information Terminals Corp.
Sunnyvale, California